



SAS[®] Viya[®]: Deployment Guide

2021.2.4*

* This document might apply to additional versions of the software. Open this document in [SAS Help Center](#) and click on the version in the banner to see all available versions.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2022. *SAS® Viya®: Deployment Guide*. Cary, NC: SAS Institute Inc.

SAS® Viya®: Deployment Guide

Copyright © 2022, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

For a hard copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

March 2022

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

v_023-P2:dplyml0phy0dkr

Contents

Chapter 1 / Pre-installation Tasks	1
Plan the Workload Placement	1
Deploy the SAS Viya Deployment Operator	7
Retrieve Required Files	13
Preparing for OpenShift	15
Directory Structure	22
Chapter 2 / Installation	25
Initial kustomization.yaml File	26
Common Customizations	29
Create the SASDeployment Custom Resource	64
Deploy the Software	69
Readiness Service	73
Sign In as the sasboot User	74
Promotion and Migration	75
Chapter 3 / Post-Installation Tasks	77
Configure Identities	78
Configure Multi-tenancy	79
Configure the Connection to the Mail Service	79
Configure Files Service	80
Configure Monitoring and Logging	80
Configure Guest Access	81
Obtain and Run Hadoop Tracer Script	81
Configure Cloud Analytic Services (CAS)	85
Configure Model Access	87
Configure SAS Data Quality	88
Configure SAS Data Science Programming	88
Configure SAS Event Stream Processing	89
Configure SAS for Microsoft 365	89
Configure SAS Intelligent Decisioning	89
Configure SAS Model Manager	90
Configure SAS Risk Modeling	90
Configure SAS Visual Analytics	90
Configure SAS Visual Data Science	91
Configure SAS Visual Data Science Decisioning	91
Configure SAS Visual Machine Learning	92
Configure SAS Visual Statistics	93
Configure SAS/CONNECT Spawner	93
Chapter 4 / Validating the Deployment	95
SAS Viya: Deployment Validation	95
SAS Viya and the SAS Operational Quality Tool	95
Chapter 5 / Modifying the Deployment Configuration	97
Modify Existing Customizations in a Deployment	97

Chapter 6 / Uninstalling	99
Uninstall with the SAS Viya Deployment Operator	99
Uninstall Manually	103
Appendix 1 / (Optional) Using a Mirror Registry	107
Create a Mirror Registry	107
Create and Populate a Mirror Registry in Microsoft Azure	114
Create and Populate a Mirror Registry in Amazon ECR	115
Create and Populate a Mirror Registry in Google Cloud Platform	116
Create a Mirror Registry at a Dark Site	118
Appendix 2 / SAS Viya Deployment Operator Fields and Messages	119
Overview	119
Fields in the SASDeployment Custom Resource	120
Manage Updates	122
Environment Variables for the Operator Pod	123
Communications from the Operator	125
Appendix 3 / PostgreSQL and Crunchy Data	129
Overview	129
Available Pods	130
Appendix 4 / Changing Deployment Methods	139
Overview	139
Deploy the SAS Viya Deployment Operator	139
Create the \$license Directory	140
Create the SASDeployment Custom Resource	140
Deploy the Software	140
Appendix 5 / Troubleshooting	141
RequestEntityTooLarge Error Message	141
Error Checking Database Connection Error Message	142
Waiting for End of Discovery Error Message	142
Certframe Container Fails at init Stage	143
Crunchy PostgreSQL Operator etcd Error Message	144
Invalid Subject Name in Partner's Certificate Error Message	145
SAS Viya Deployment Operator Reconcile Failures	146
Performing Updates Produces Many Error Messages	146
Failure to Create Symbolic Link	146
Long Delay for SAS/CONNECT Server Sign-on from SAS Studio	147

Pre-installation Tasks

Plan the Workload Placement	1
Introduction	2
Assign Nodes by Class	2
Failures When Adding Workload Node Placement	7
Deploy the SAS Viya Deployment Operator	7
Overview of the SAS Viya Deployment Operator	7
Retrieve the Files Required by the SAS Viya Deployment Operator	8
Edit the transformer.yaml File	10
Edit the kustomization.yaml for Cluster-Wide Mode	11
Edit the kustomization.yaml for seccomp	11
Configure the Mirror Registry	11
Add an imagePullSecret	12
Apply the SAS Viya Deployment Operator Resources to the Cluster	13
Retrieve Required Files	13
Preparing for OpenShift	15
Workload Node Placement Considerations	15
Security Context Constraints and Service Accounts	15
Enable hostPath Mounts for CAS	20
Networking	20
TLS	21
Additional Security	21
Directory Structure	22
\$deploy/sas-bases Directory	22
\$deploy/site-config Directory	23
Directories for the SAS Viya Deployment Operator	24

Plan the Workload Placement

Note: If you have used a GitHub project, SAS Viya Infrastructure as Code (IaC) for one of the cloud providers, to provision your infrastructure components, the

procedures described in this section have already been performed for you. You might still want to verify the taints and labels with the following command:

```
kubect1 get nodes -o=custom-  
columns=NAME:.metadata.name,LABELS:.metadata.labels,TAINTS:.spec.taints
```

For more information about commands to validate the workload node placement in your infrastructure, see [“Place the Workload on Nodes” on page 6](#). For more information about IaC, see [“Help with Cluster Setup” in System Requirements for SAS Viya](#).

Introduction

Before you deploy your SAS Viya software, you should develop a plan to distribute the workload across your deployment. Labeling and tainting the nodes prepares the environment to use your plan when the manifest is applied during the deployment process. This section explains how to distribute the workload across your deployment according to several factors.

Assign Nodes by Class

Overview

SAS Viya consists of multiple workload classes. Each class of workload has a unique set of attributes that you must consider when planning your deployment. When planning the placement of the workload, it is helpful to think beyond the initial deployment and to also consider the Kubernetes maintenance life cycle.

IMPORTANT SAS strongly recommends labeling and tainting all your nodes, especially the CAS nodes.

Workload Classes

Properties associated with the pods in your deployment describe the type of work that they perform. By using the commands described in [“Place the Workload on Nodes” on page 6](#), you can configure where you want each class of pods to run, thereby enabling you to manage the associated workload.

Here are the SAS Viya workload classes:

stateless

These workloads include web applications and microservices. Stateless applications scale horizontally in order to handle increased workload and to

provide higher availability. By default, one replica of each stateless workload is deployed. A high-availability (HA) scaling overlay is provided with the deployment artifacts for each SAS Viya software order.

stateful

These workloads include open-source components that are used by SAS Viya in order to store critical information that is required to maintain operational integrity. These workloads are deployed in an HA configuration by default.

cas

SAS Cloud Analytic Services (CAS) is the primary compute engine in SAS Viya. Although CAS is a stateful process, it has unique attributes and requirements within the Kubernetes infrastructure. CAS has been engineered to use all available resources to complete each request in the shortest amount of time. Therefore, CAS can quickly drive every available CPU core to 100% utilization. CAS also aggressively consumes RAM to process large volumes of data as quickly as possible.

Because CAS was designed with performance as a primary objective, it must be hosted on dedicated resources. Therefore, each CAS pod must be placed on its own dedicated node. In addition, CAS must be configured with guaranteed quality of service (QoS) in order to prevent CAS pods from being evicted by Kubernetes.

Note: The requirement to place each CAS pod on its own dedicated node does not apply to the personal CAS server. For more information about the personal CAS server, see [“Create a Personal CAS Server” on page 57](#).

compute

Compute processes are stateful processes that run to completion (that is, they are not long-running services). They deliver multiple functions within a SAS Viya deployment, including but not limited to batch jobs, interactive SAS Studio sessions, and parallel compute processes that are executed as part of a SAS Model Studio workflow.

connect

SAS/CONNECT provides the ability for clients to launch SAS compute processes, the same processes described in [“compute” on page 3](#), within a local or remote SAS environment. The SAS/CONNECT Spawner is a SAS Viya service that launches these processes on behalf of SAS/CONNECT clients. The processes can be launched in their own pods (referred to as “dynamically launched pods”) or in the SAS/CONNECT Spawner pod. SAS 9 clients from releases before SAS 9.4M7 and SAS Viya 3 clients from versions before SAS Viya 3.5 do not support dynamically launched pods, so by default their processes are launched in the SAS/CONNECT Spawner pod. For other clients, dynamically launched pods are enabled.

If you are using external SAS/CONNECT clients with your SAS Viya deployment, the SAS/CONNECT Spawner requires more resources in order to avoid being evicted by Kubernetes. Like CAS, the SAS/CONNECT Spawner pod requires a dedicated Kubernetes node and must be configured with guaranteed QoS.

Default Node Pool Configuration

SAS recommends that you use a node pool that does not run SAS workloads in the cluster. This node pool is referred to as the *default node pool*. The default node pool is useful for ingress controllers and for hosting LDAP and monitoring stacks. SAS also recommends that the default node pool be labeled with `kubernetes.azure.com/mode=system` so that SAS pods are not scheduled there.

Note: This label is automatically applied to Microsoft Azure Kubernetes Service system node pools. If you are not provisioning the cluster with the SAS-provided IaC tooling, the label may need to be applied to the equivalent default node pool for other infrastructures.

To apply the label to default nodes or node pools in other infrastructures, use the following command:

```
kubectl label nodes node-name kubernetes.azure.com/mode=system
```

Note: The *node-name* can be the name of a single node or it can be a list of node names separated by a space. The term `azure` is applied to all of the supported cloud platforms because it represents a default SAS Viya deployment setting that is not vendor-specific.

Considerations and Default Settings for Workload Placement

The Kubernetes cluster administrator will recognize that each of the SAS Viya workload classes should be managed differently. In situations where the deployment has rigorous availability requirements and the Kubernetes cluster software is being upgraded, the different requirements might resemble the following:

- Nodes that host the stateless and stateful workload class can be individually drained and updated with no additional management concerns as long as the HA transformer was used for the deployment.
- Nodes that host the compute workload class should be cordoned. The workloads that run on those nodes should be allowed to run to completion in order to avoid interrupting work in progress. Examples are a SAS Studio session, a batch job, or a SAS Model Studio session. When all compute processes have completed, the node can be drained so that maintenance tasks can be performed.

Each node that hosts compute class workloads can be cordoned, drained, or updated in a manner that does not disrupt end-user activity as long as each compute session is allowed to run to completion.

Note: If your deployment includes SAS Workload Management, you must have at least one node labeled for the compute workload class.

- Nodes that host workloads for the cas and connect classes must be terminated and started on different nodes before the underlying nodes can be maintained as required.

Some aspects of workload placement are set by default during the SAS Viya deployment, including both nodeAffinity and tolerations. In addition, SAS Viya workloads are not scheduled on the system node pool in Microsoft Azure, the default node group in Amazon Web Services (AWS), or the default node pool in Google Cloud Platform (GCP).

The following table summarizes additional implications of the SAS Viya default deployment for each workload class:

Table 1.1 Default Taints and nodeAffinity Settings per Class

Workload Class	Default Settings
Stateless workloads	<p>Prefer to schedule on nodes that are labeled workload.sas.com/class=stateless</p> <p>Tolerate the following taints:</p> <ul style="list-style-type: none"> ■ workload.sas.com/class=stateless:NoSchedule ■ workload.sas.com/class=stateful:NoSchedule
Stateful workloads	<p>Prefer to schedule on nodes that are labeled workload.sas.com/class=stateful</p> <p>Tolerate the following taints:</p> <ul style="list-style-type: none"> ■ workload.sas.com/class=stateful:NoSchedule ■ workload.sas.com/class=stateless:NoSchedule
Compute workloads	<p>Prefer to schedule on nodes that are labeled workload.sas.com/class=compute</p> <p>Tolerate the taint workload.sas.com/ class=compute:NoSchedule</p>
CAS workloads	<p>Prefer to schedule on nodes that are labeled workload.sas.com/class=cas</p> <p>Tolerate the taint workload.sas.com/ class=cas:NoSchedule</p>
Connect workloads	<p>Prefer to schedule on nodes that are labeled workload.sas.com/class=connect</p> <p>Tolerate the taint workload.sas.com/ class=connect:NoSchedule</p>

If a CAS workload is scheduled to run on a specific node, no other application pods can be scheduled to run on that node. However, the CAS operator can be used to perform additional configuration. For example, the nodes that run CAS workloads require guaranteed QoS. Therefore, the CAS operator can configure the CAS manifest to enable guaranteed QoS on these nodes. For more information, see the README file located at `sas-bases/overlays/cas-server/auto-resources/`

`README.md` (for Markdown format) or `$deploy/sas-bases/docs/auto_resources_for_cas_server_for_sas_viya.htm` (for HTML format).

If you deploy SAS Viya into a cluster with untainted nodes, the Kubernetes scheduler may attempt to schedule CAS onto those untainted nodes instead of the nodes that have been labeled with `workload.sas.com/class=cas`. An overlay that ensures that CAS servers are only scheduled onto nodes that have the dedicated label is available. For more information, see the “Optional CAS Server Placement Configuration” section of the README file located at `$deploy/sas-bases/overlays/cas-server/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/mpp_cas_server_for_sas_viya.htm` (for HTML format).

Multi-tenancy

If you are enabling multi-tenancy in your deployment, the tenants will share most of the nodes with the provider tenant. However, because each tenant has its own CAS server, the total number of nodes required for CAS for the full deployment is greater than that for a non-multi-tenant deployment. The number of additional CAS nodes required per tenant depends on whether the tenant is deploying SMP or MPP CAS.

Place the Workload on Nodes

SAS requires that you identify the node or nodes on which CAS pods should be scheduled. SAS further recommends that you identify the nodes on which all classes should be scheduled. The following commands place the workload on the nodes in your deployment:

To list the names of the nodes:

```
kubectl get nodes
```

To assign a class of pods to a specific node:

```
kubectl taint nodes node-name workload.sas.com/class=class-name:NoSchedule --overwrite
kubectl label nodes node-name workload.sas.com/class=class-name --overwrite
```

Note: The *node-name* can be the name of a single node or it can be a list of node names separated by a space.

Here is an example that assigns only CAS pods to a node named `node1`.

```
kubectl taint nodes node1 workload.sas.com/class=cas:NoSchedule --overwrite
kubectl label nodes node1 workload.sas.com/class=cas --overwrite
```

To verify the nodes that have been tainted:

```
kubectl get nodes -o=custom-columns=NAME:.metadata.name,TAINTS:.spec.taints
```

To view node labels and taints at the same time:

Note: Because of the length of the command and the margin of the page, this command appears as more than one line. The command should be entered as a single line.

```
kubectl get nodes -o=custom-  
columns=NAME:.metadata.name,LABELS:.metadata.labels,TAINTS:.spec.taints
```

After you have assigned the pods to nodes, to check the assignments:

Note: Because of the length of the command and the margin of the page, this command appears as more than one line. The command should be entered as a single line.

```
kubectl -n name-of-namespace get pod -o=custom-  
columns=NAME:.metadata.name,NODE:.spec.nodeName | sort
```

Failures When Adding Workload Node Placement

If you update from an environment without workload node placement to one that uses workload placement, you might encounter scheduling failures with the `sas-crunchy-data-postgres-rmdata` container. If those errors occur, run the following commands to delete the PostgreSQL cluster without deleting any of the data that it contains:

```
kubectl -n name-of-namespace delete deployment -l vendor=crunchydata  
kubectl -n name-of-namespace delete service -l vendor=crunchydata  
kubectl -n name-of-namespace delete cm -l vendor=crunchydata  
kubectl -n name-of-namespace delete job -l vendor=crunchydata  
kubectl -n name-of-namespace delete pgcluster --all
```

Deploy the SAS Viya Deployment Operator

Overview of the SAS Viya Deployment Operator

The SAS Viya Deployment Operator provides an automated method for deploying and updating your SAS Viya deployment. It runs in the Kubernetes cluster and watches for declarative representations of SAS Viya deployments in the form of custom resources (CRs) of the type `SASDeployment`. The operator can watch for `SASDeployments` in the namespace where it is deployed (namespace mode) or in all the namespaces in a cluster (cluster-wide mode). Using namespace mode means that the operator and the SAS Viya deployment are located in the same namespace. Operators in cluster-wide mode have their own namespaces.

Note: SAS recommends using only one mode of operator in a cluster.

When a new `SASDeployment` is created or an existing `SASDeployment` is updated, the operator updates the SAS Viya deployment to match the state that is described

in the CR. The operator determines the release of SAS Viya that it is working with. It also uses the appropriately versioned tools for that release while deploying and updating the SAS Viya software. Thus, a single instance of the operator can manage all SAS Viya deployments in the cluster.

The SAS Viya Deployment Operator requires updates occasionally but not at the same frequency as SAS Viya.

If you want to consider using the SAS Viya Deployment Operator to manage your SAS Viya deployments, continue reading this section for its deployment steps. If you want to deploy and update your software manually, skip this section and go to [“Retrieve Required Files” on page 13](#).

Retrieve the Files Required by the SAS Viya Deployment Operator

Note: If you plan to use a mirror registry for the deployment of your SAS Viya software, you should download and install SAS Mirror Manager before you begin your deployment. For more information, see [“Create a Mirror Registry” on page 107](#).

The required deployment assets are delivered in a TGZ file that is accessible from the my.sas.com portal.

- 1 If you plan to use a cluster-wide mode SAS Viya Deployment Operator and you have not already created a namespace that is dedicated to the deployment of the SAS Viya Deployment Operator, create that namespace now. If you plan to use a namespace mode SAS Viya Deployment Operator and you have not created a namespace for your SAS Viya deployment, create that namespace now.

Note: This namespace is referred to as `sasoperator` in the examples that follow.

- 2 Create a directory on the kubectl machine or on a machine that can be reached by your kubectl machine:

```
mkdir directory-name
```

Note: If you created a mirror registry, the new directory should be parallel to \$deploy.

SAS recommends that you name the directory `operator-deploy`, but you should use a name that is meaningful to you. The directory is referred to as \$operator-deploy in this guide. Replace \$operator-deploy with the directory name that you prefer.

- 3 Click the **Get Started** link that is provided in your Software Order Email (SOE). The my.sas.com page opens.
- 4 Expand the information for the order that is listed in your SOE.

- 5 In the pane that opens, examine the order information. The version indicates the release cadence and the version of SAS Viya software that you will deploy. If you want to deploy a different version, select the cadence and release from the **SAS Viya Version** list.

Note: If you are changing your deployment method from manual to operator prior to performing an update, be sure to download the deployment assets for the version you intend to update to. The target version should be used for deploying the deployment operator.

- 6 Click **Download Deployment Assets** to download the files that are required to deploy your software.
- 7 Save the TGZ file from the my.sas.com page to the directory that you created in step 2.
- 8 Extract the files from the TGZ file in the same directory:

```
tar xvfz file-name.tgz
```

The result is a directory structure that looks like this:

```
$operator-deploy/
├─ sas-bases/
│   ├── base/
│   ├── docs/
│   ├── examples/
│   ├── extras/
│   └─ overlays/
```

- 9 Copy the operator files to the top level of the \$operator-deploy directory and make them writable:

```
cd $operator-deploy
cp -r sas-bases/examples/deployment-operator/deploy/* .
chmod +w site-config/transformer.yaml
```

The result is a directory structure that looks like this:

```
$operator-deploy/
├─ kustomization.yaml
├─ operator-base/
├─ sas-bases/
└─ site-config/
    ├── cluster-wide-transformer.yaml
    └─ transformer.yaml
```

The kustomization.yaml file in the \$operator-deploy directory is referred to as the operator kustomization.yaml file throughout the documentation.

Note: Verify that you are using the correct directory for your changes: `$operator-deploy/site-config` OR `$deploy/site-config`.

Edit the transformer.yaml File

Replace the variables in the `$operator-deploy/site-config/transformer.yaml` file as indicated in that file. Two patches must be updated.

- The first patch sets the name of the ClusterRoleBinding resource that is needed by the operator. The SAS Viya Deployment Operator requires cluster-wide permissions even when the operator is deployed in namespace mode because the SAS Viya deployment requires the deployment of cluster resources. Because ClusterRoleBinding is a cluster-wide resource, it must be unique within the cluster. Each SAS Viya Deployment Operator instance must have a unique ClusterRoleBinding.
- The second patch sets the name of the namespace in which the operator is being deployed, and therefore where the RBAC resources, such as ServiceAccount, are deployed

Here is an example of the transformer.yaml file. It contains the name *sasoperator* for the ClusterRoleBinding and the namespace *sasoperator*.

Note: If you are deploying the operator in namespace mode, the namespace value is also used in the base kustomization.yaml file since you are also deploying SAS Viya into it.

```
---
apiVersion: builtin
kind: PatchTransformer
metadata:
  name: patch-transformer-sas-deployment-operator
patch: |-
  # name: MUST BE PROVIDED BY USER AND UNIQUE IN CLUSTER.
  - op: replace
    path: /metadata/name
    value:
      "sasoperator"
  # namespace: MUST BE PROVIDED BY USER. THIS MUST BE THE NAMESPACE INTO WHICH
  # THE SAS DEPLOYMENT OPERATOR IS DEPLOYED.
  - op: add
    path: /subjects/0/namespace
    value:
      "sasoperator"
target:
  annotationSelector: sas.com/component-name=sas-deployment-operator
  # Some required components for SAS Viya are scoped to the cluster. In order to
  # deploy your software, the operator requires the ClusterRoleBinding, regardless
  # of whether the intended scope is to the namespace or the cluster.
  kind: ClusterRoleBinding
```

Edit the kustomization.yaml for Cluster-Wide Mode

If the operator is being deployed in namespace mode, skip this section. However, if the operator is being deployed in cluster-wide mode, edit the operator kustomization.yaml to uncomment the reference to site-config/cluster-wide-transformer.yaml. Here is an example of the file when deploying in cluster-wide mode.

```
resources:
- operator-base
transformers:
- site-config/transformer.yaml
## Uncomment the following inclusion if you are deploying the
## operator in cluster-wide mode.
- site-config/cluster-wide-transformer.yaml
```

Edit the kustomization.yaml for seccomp

If you are using the operator in a cluster that does not support secure computing mode (seccomp), uncomment the relevant line in the operator kustomization.yaml file. Red Hat OpenShift does not support seccomp.

Here is an example:

```
transformers:
...
## Uncomment the following inclusion if you are deploying the
## operator in a cluster that does not support seccomp.
- site-config/remove-seccomp-transformer.yaml
```

Configure the Mirror Registry

Note: If you not using a mirror registry for your deployment, skip this section.

- 1 Copy the mirror.yaml file from `$operator-deploy/sas-bases/examples/mirror/` to the `$operator-deploy/site-config` directory.
- 2 Open the mirror.yaml file and replace each instance of `{{ MIRROR-HOST }}` with the fully qualified domain name (FQDN) of the mirror registry and the registry namespace in this format:

`FQDN/registry-namespace`

Here is an example:

`example.company.com/SASdeploy`

After replacing all instances, save and close the file.

- 3 In the operator `kustomization.yaml` file, add the path to the `mirror.yaml` file as the last entry in the `transformers` section. For example, if you copied the `mirror.yaml` file to the top level of the `$operator-deploy/site-config` directory, the entry would look like this:

```
transformers:
- site-config/transformer.yaml
- site-config/mirror.yaml
## Uncomment the following inclusion if you are deploying the
...
```

Add an imagePullSecret

Note: If you are not using a mirror registry or if you are using a mirror registry but authentication is not required, skip this section.

If SAS content has been mirrored, and that mirror requires authentication, you must configure an `imagePullSecret`. See [Pull an Image from a Private Registry](#) for information about creating an `imagePullSecret`. The secret can be added to the SAS Viya Deployment Operator's Deployment resource in the operator `kustomization.yaml` file. Here is an example of the resulting `kustomization.yaml` file that assumes that the secret content is in the `site-config/image-pull-secret.json` file:

```
resources:
- operator-base
transformers:
- site-config/transformer.yaml
## Uncomment the following inclusion if you are deploying the
## operator in clusterwide mode.
- site-config/cluster-wide-transformer.yaml
secretGenerator:
- name: site-image-pull-secret
  type: kubernetes.io/dockerconfigjson
  files:
  - .dockerconfigjson=site-config/image-pull-secret.json
patches:
- patch: |-
  - op: add
    path: /spec/template/spec/imagePullSecrets
    value:
    - name: site-image-pull-secret
target:
  kind: Deployment
  name: sas-deployment-operator
```


Apply the SAS Viya Deployment Operator Resources to the Cluster

Run the following command from the `$operator-deploy` directory to deploy the SAS Viya Deployment Operator into Kubernetes:

```
kustomize build . | kubectl -n name-of-deployment-operator-namespace apply -f -
```

Retrieve Required Files

Note: If you plan to use a mirror registry for the deployment of your SAS Viya software and you have not already done so, you should download and install SAS Mirror Manager before beginning your deployment. For more information, see [“Create a Mirror Registry” on page 107](#).

- 1 Deploying SAS Viya requires a separate namespace for each deployment of your software. If you are using the SAS Viya Deployment Operator in namespace mode, the SAS Viya deployment must be performed in the same namespace that was used for the operator.

If you have not already done so, create a namespace for your SAS Viya deployment.

- 2 Create the directory for your deployment assets.

- If you have created a mirror registry, you have already created the `$deploy` directory and should skip to step 3.
- Create a directory on the `kubectl` machine or on a machine that can be reached by your `kubectl` machine:

```
mkdir directory-name
```

Note: If you have deployed the SAS Viya Deployment Operator, the new directory should be parallel to `$operator-deploy`.

SAS recommends that you name the directory `deploy`, but you should use a name that is meaningful to you. The directory is referred to as `$deploy` in this guide. Replace `$deploy` with the directory name that you prefer.

If you are performing multiple deployments of SAS Viya in a cluster, each `$deploy` directory should be named in a way that allows you to associate a directory with a specific deployment.

Note: If you are deploying with the SAS Viya Deployment Operator, you can use a GitOps repository to store your deployment assets rather than create a

directory. If you choose to use GitOps, the instructions for \$deploy throughout the documentation apply to the GitOps directory as well.

- 3 Click the **Get Started** link that is provided in your Software Order Email (SOE). The my.sas.com page opens.
- 4 Expand the information for the order that is listed in your SOE.
- 5 In the pane that opens, examine the order information. The version indicates the release cadence and the version of SAS Viya software you are about to deploy. If you want to deploy a different version, select the cadence and release from the **SAS Viya Version** list.

- 6 If you are performing a manual deployment, skip this step.
Retrieve your certificates and license to deploy your software.

- a Create a new directory parallel to the \$deploy directory in which to store your license and *-certs.zip files.

```
mkdir directory-name
```

You should use a name that is meaningful to you. The directory is referred to as \$license in this guide. Replace \$license with the directory name that you prefer.

- b Click **Download Certificates** and download the *-certs.zip file. Place the .zip file in the \$license directory.
- c Click **Download License Only** to download the license file, identified by the .jwt extension. Save the .jwt file in the \$license directory.

Note: If you plan to have multiple SAS Viya deployments in your cluster, you should organize the \$license directory as you see fit. Whatever strategy you use to organize, ensure that you can easily differentiate the license and certs.zip files by order.

- 7 Click **Download Deployment Assets** to download the files required to deploy your software.
- 8 Save the TGZ file from the my.sas.com page to the directory that you created in step 2.
- 9 Extract the files from the TGZ file in the same directory:

```
tar xvfz file-name.tgz
```

The result is a directory structure that looks like this:

```
$deploy/
├─ sas-bases/
│   ├── base/
│   ├── components/
│   ├── docs/
│   ├── examples/
│   └─ overlays/
```

For a description of each subdirectory, see “[\\$deploy/sas-bases Directory](#)” on [page 22](#).

- 10 Parallel to the `sas-bases` subdirectory, create a new directory named `site-config`.

For a description of the `$deploy/site-config` directory, see “[\\$deploy/site-config Directory](#)” on page 23.

Preparing for OpenShift

Note: If you are not deploying SAS Viya in a Red Hat OpenShift environment, skip this section.

Deployment steps for OpenShift are different from those required for other infrastructures. This section describes those differences.

Workload Node Placement Considerations

Although the workload placement documentation recommends labelling and tainting all worker nodes, you must have at least two untainted default nodes to allow for the default ingress controller replica count. For more information, see [Ingress controller configuration parameters](#).

Security Context Constraints and Service Accounts

A deployment in OpenShift requires multiple custom security context constraints (SCCs) to provide permissions to SAS Viya services. SCCs are required in order to enable the Pods to run. (Pods provide essential SAS Viya components.) In addition, some SCCs can be customized to meet your unique requirements.

A security context acts like a request for privileges from the OpenShift API. In an OpenShift environment, each Kubernetes Pod starts up with an association with a specific SCC, which limits the privileges that Pod can request. An administrator configures each Pod to run with a certain SCC by granting the corresponding service account for that pod access to the SCC. For example, if Pod A requires its own SCC, an administrator must grant access to that SCC for the service account under which Pod A is launched. Use the OpenShift OC administrative command-line tool to grant or remove these permissions.

- 1 After you have downloaded the deployment assets as described in “[Retrieve Required Files](#)” on page 13, apply the following SCCs that pertain to your deployment and then bind the SCC to the appropriate service account.

Note: For additional details about SCC types, see “[SCCs and Pod Service Accounts](#)” in *System Requirements for SAS Viya*.

■ **sas-cas-server**□ **Command to apply the SCC:**

```
kubectl apply -f sas-bases/examples/cas/configure/cas-server-scc.yaml
```

or

```
kubectl apply -f sas-bases/examples/cas/configure/cas-server-scc-host-launch.yaml
```

□ **Command to bind the SCC to the service account:**

```
oc -n name-of-namespace adm policy add-scc-to-user sas-cas-server -z sas-cas-server
```

or

```
oc -n name-of-namespace adm policy add-scc-to-user sas-cas-server-host -z sas-cas-server
```

- **Why the SCC is needed:** CAS relies on SETUID, SETGID, and CHOWN capabilities. CAS is launched by a SETUID root executable called caslaunch. By default, caslaunch starts the CAS controller running under the runAsUser/runAsGroup values and a root process named launchsvcs. Caslaunch connects these processes with a pipe. The SETUID and SETGID capabilities are required by launchsvcs in order to launch session processes under the user's operating-system (host) identity instead of launching them using runAsUser/runAsGroup values. The CHOWN capability is necessary to support Kerberos execution, which requires modification of the ownership of the cache file that is created by a direct Kerberos connection. By default, the cache file is owned by runAsUser/runAsGroup identities, but in order to support Kerberos, it must be owned by the user's host identity.

For more information, including which SCC to use, see the README file at `$deploy/sas-bases/overlays/cas-server/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/cas_server_for_sas_viya.htm` (for HTML format).

■ **sas-connect-spawner**

Note: By default, no SCC is required for SAS/CONNECT and you can skip this item. The SCC is required only if you intend to launch your SAS/CONNECT servers in the Spawner pod, rather than in their own pods. For more information, see the README described below.

□ **Command to apply the SCC:**

```
kubectl apply -f sas-bases/examples/sas-connect-spawner/openshift/sas-connect-spawner-scc.yaml
```

□ **Command to bind the SCC to the service account:**

```
oc -n name-of-namespace adm policy add-scc-to-user sas-connect-spawner -z sas-connect-spawner
```

- **Why the SCC is needed:** The SAS/CONNECT Launcher must be able to launch the SAS/CONNECT Server under end user identity.

For more information, see the README file at `$deploy/sas-bases/examples/sas-connect-spawner/openshift/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/`

`granting_security_context_constraints_on_an_openshift_cluster_for_r_sasconnect.htm` (for HTML format).

■ pgo (internal instance of PostgreSQL)

Note: If your deployment will use an external instance of PostgreSQL, skip this item.

□ Command to apply the SCC:

```
kubectl apply -f sas-bases/examples/crunchydata/openshift/pgo-scc.yaml
```

□ Commands to bind the SCC to the service accounts:

```
oc -n name-of-namespace adm policy add-scc-to-user pgo -z pgo-pg
```

and

```
oc -n name-of-namespace adm policy add-scc-to-user pgo -z pgo-target
```

- Why the SCC is needed: After mounting the storage class via PVC, the Crunchy Data database container changes the ownership of all the files that the database process writes to a uid of 26 and a gid of 26. Since it is a third-party product, there is no way for SAS to disable the file ownership.

For more information, see the README file at `$deploy/sas-bases/examples/crunchydata/openshift/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/granting_security_context_constraints_on_an_openshift_cluster.htm` (for HTML format).

■ pgo-backrest

Note: If your deployment will use an external instance of PostgreSQL, skip this item.

□ Command to apply the SCC:

```
kubectl apply -f sas-bases/examples/crunchydata/openshift/pgo-backrest-scc.yaml
```

□ Commands to bind the SCC to the service accounts:

```
oc -n name-of-namespace adm policy add-scc-to-user pgo-backrest -z pgo-default
```

and

```
oc -n name-of-namespace adm policy add-scc-to-user pgo-backrest -z pgo-backrest
```

- Why the SCC is needed: After mounting the storage class via PVC, the Crunchy Data pgbbackrest container changes the ownership of all the files that the database process writes to a uid of 2000 and a gid of 2000. Since it is a third-party product, there is no way for SAS to disable the file ownership.

For more information, see the README file at `$deploy/sas-bases/examples/crunchydata/openshift/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/`

`granting_security_context_constraints_on_an_openshift_cluster.htm` (for HTML format).

■ sas-opendistro

- Deploying Open Distro for Elasticsearch on OpenShift requires changes to a few kernel settings. The `sysctl-transformer.yaml` file can apply the necessary `sysctl` parameters to configure the kernel, but it requires the following special privileges:

- `allowPrivilegeEscalation` option enabled
- `allowPrivilegedContainer` option enabled
- `runAsUser` set to `RunAsAny`

Therefore, before you apply the `sas-opendistro-scc.yaml` file, you must modify it to enable these privileges. For the instructions to modify them, see the “Modify `sas-opendistro-scc.yaml` for `sysctl-transformer.yaml`” section of the README file at `$deploy/sas-bases/examples/configure-elasticsearch/internal/openshift/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/open_distro_for_elasticsearch_on_red_hat_openshift.htm` (for HTML format).

- Command to apply the SCC:

```
kubectl apply -f sas-bases/examples/configure-elasticsearch/internal/openshift/sas-opendistro-scc.yaml
```

- Command to bind the SCC to the service account:

```
oc -n name-of-namespace adm policy add-scc-to-user sas-opendistro -z sas-opendistro
```

- Why the SCC is needed: For optimal performance, Open Distro for Elasticsearch software requires the `vm.max_map_count` parameter to be set on the Kubernetes nodes running the stateful workloads to ensure there is adequate virtual memory available for use with `mmap` for accessing the search indices. To provide a method to set this argument automatically, SAS Viya includes an optional transformer as part of the `internal-elasticsearch` overlay that adds an `init` container to automatically set this parameter. The `init` container must be run at a privileged level (using both the `privileged = true` and `allowPrivilegeEscalation = true` security context options) since it modifies the kernel parameters of the host. The container terminates after it sets the kernel parameter, it terminates, and the Open Distro for Elasticsearch software will then proceed to start as a non-privileged container.

■ sas-microanalytic-score

- Command to apply the SCC:

```
kubectl apply -f sas-bases/overlays/sas-microanalytic-score/service-account/sas-microanalytic-score-scc.yaml
```

- Command to bind the SCC to the service account:

```
oc -n name-of-namespace adm policy add-scc-to-user sas-microanalytic-score -z sas-microanalytic-score
```

- Why the SCC is needed: SAS Microanalytic Score can leverage Python through the mounting of NFS. NFS volumes are not permitted in the restricted SCC, so an SCC that has NFS in the allowed volumes section is required.

For more information, see the README file at `$deploy/sas-bases/overlays/sas-microanalytic-score/service-account/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_sas_micro_analytic_service_to_add_service_account.htm` (for HTML format).

■ SAS Watchdog

- Command to apply the SCC:

```
kubectl apply -f sas-bases/examples/sas-programming-environment/watchdog/sas-watchdog-scc.yaml
```

- Command to bind the SCC to the service account:

```
oc -n name-of-namespace adm policy add-scc-to-user sas-watchdog -z sas-programming-environment
```

- Why the SCC is needed: SAS Watchdog monitors spawned processes to ensure that they comply with the terms of LOCKDOWN system option. It emulates the restrictions imposed by LOCKDOWN by restricting access only to files that exist in folders that are allowed by LOCKDOWN. It therefore requires elevated privileges provided by the custom SCC.

For more information, see the README file at `$deploy/sas-bases/overlays/sas-programming-environment/watchdog/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configuring_sas_compute_server_to_use_sas_watchdog.htm` (for HTML format).

- 2 If you plan to publish models with SAS Model Manager or SAS Intelligent Decisioning using kaniko, you must bind the `sas-model-publish-kaniko` service account to the `anyuid` SCC:

```
oc -n name-of-namespace adm policy add-scc-to-user anyuid -z sas-model-publish-kaniko
```

Note: The `anyuid` SCC is a standard SCC defined by OpenShift. For more information about the `anyuid` SCC, see [Managing SCCs in OpenShift](#).

For more information, see the README at `$deploy/sas-bases/examples/sas-model-publish/kaniko/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_kaniko_for_sas_model_publish_service.htm` (for HTML format).

- 3 The SAS Watchdog service account includes the same permissions as the service account in this step. If you have already bound the SAS Watchdog service account as described in step 1, you should skip this step.

Bind the `hostmount-anyuid` or `anyuid` SCC to the `sas-programming-environment` service account using this command:

```
oc -n name-of-namespace adm policy add-scc-to-user hostmount-anyuid -z sas-programming-environment
```

Using the `anyuid` SCC is preferred. Using the `hostmount-anyuid` SCC is only required if you use `hostPath` mounts.

Note: The `hostmount-anyuid` and `anyuid` SCCs are standard SCCs defined by OpenShift. For more information about them, see [Managing SCCs in OpenShift](#).

Enable hostPath Mounts for CAS

If you want to use hostPath mounts for CAS in your OpenShift deployment, perform the following steps:

- 1 Add hostPath to the list of volume types in the SCC being used. Here is an example:

```
volumes:
- configMap
- downwardAPI
- emptyDir
- persistentVolumeClaim
- projected
- secret
- nfs
- hostPath
```

The SCC to modify is `cas-server-scc.yaml` or `cas-server-scc-host-launch.yaml` if you are enabling host launch for CAS.

- 2 Run the following command on the CAS worker node with the directory to be used as a hostPath:

```
sudo chcon -t container_file_t -R directory-path
```

Replace *directory-path* with the directory path to be used as a hostPath mount. This directory must not be located in the file system's root directory (/).

Note: If you prefer, you can use a MachineConfig to apply the same chcon change instead. Consult your cluster administrator to determine which option is best for your deployment.

Networking

OpenShift uses *routes* for its ingress controllers. The initial kustomization.yaml file does not account for routes, so you must change the resources block. Here is the original kustomization.yaml content:

```
resources:
- sas-bases/base
- sas-bases/overlays/cert-manager-issuer
- sas-bases/overlays/network/networking.k8s.io
```

Here is the revision for OpenShift:

```
resources:
- sas-bases/base
- sas-bases/overlays/cert-manager-issuer
- sas-bases/overlays/network/route.openshift.io
```


For details, see the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

TLS

Because OpenShift uses routes instead of Ingress, the security component listed in the initial kustomization.yaml file must be revised. Here is the original kustomization.yaml content:

```
components:
- sas-bases/components/security/core/base/full-stack-tls
- sas-bases/components/security/network/networking.k8s.io/ingress/
  nginx.ingress.kubernetes.io/full-stack-tls
```

Here is the revision for OpenShift:

```
components:
- sas-bases/components/security/core/base/full-stack-tls
- sas-bases/components/security/network/route.openshift.io/route/full-
  stack-tls
```

For details, see the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

Additional Security

You must perform two additional tasks to apply security for OpenShift:

- remove the Secure Computing (seccomp) profile
- update the fsGroup field

For the instructions to perform both tasks, see the README file at `$deploy/sas-bases/examples/security/container-security/README.md` (for Markdown format) or `$deploy/sas-bases/docs/modify_container_security_settings.htm` (for HTML format).

Directory Structure

\$deploy/sas-bases Directory

The `$deploy/sas-bases` directory contains the files that are provided by SAS to deploy your software. Here is the structure:

```
sas-bases/  
├─ base/  
├─ components/  
├─ docs/  
├─ examples/  
├─ extras/  
└─ overlays/
```

The examples subdirectory contains files that are organized by categories, such as CAS and security, that control the configuration of your deployment. README.md files describe specific configuration settings and the files and modifications that are required to enable those settings. Example files contain the variables that control configuration settings, such as the number of workers in an MPP deployment of CAS. If you decide to enable the configuration setting that is described in a README.md file, you should copy the example file that is listed in the README.md file to the site-config directory, and replace the variables with the values that you want for your deployment.

The overlays subdirectory also contains files, organized by category, that affect the configuration of your deployment. However, unlike the examples directory, the files in the overlays directory do not need to be modified before they are used. Therefore, they also do not have to be copied to the site-config directory. To use an overlay, you add a reference to the specific file to your base kustomization.yaml file.

The docs subdirectory contains HTML versions of the README files in your software order with an HTML index for them.

Depending on the software in your order, you might have an extras subdirectory. It contains configuration files for special deployment options. It is also subdivided into examples and overlays subdirectories that act in the same way as the higher level examples and overlays directories.

The base and components subdirectories do not contain any files that you should revise.

\$deploy/site-config Directory

Overview

If you are following the instructions in this guide, you have created a directory structure for your SAS Viya deployment that looks like this:

```
$deploy/  
├─ sas-bases/  
└─ site-config/
```

The `$deploy/sas-bases` directory contains the files that SAS provides to deploy your software. During the process of deploying your software, you will be instructed to edit files in the `sas-bases` directory. You should not edit these files directly. Instead, copy the file to the `$deploy/site-config` directory, ensure that the copied file is writeable, and make your modifications to the copied file.

You are being directed to modify copied files because future updates of your software involving replacing the `sas-bases` directory with an updated one. If you store modified files that control the installation and configuration of your software in `sas-bases`, those files are lost during the update. Updates do not change the files in `$deploy/site-config`, although you might be asked to modify a file that exists there.

Organization

Because the `$deploy/site-config` directory contains files that you modify, you should organize the directory according to your needs. However, it is important that you plan the organization before you continue the deployment process. You should organize your files in the `site-config` directory so that you and others who work on the deployment can locate them easily.

Because the structure that you choose for your `site-config` directory depends on your choices, explicit instructions on where to save specific files cannot be provided. Instead, only general instructions can be offered. An example of a general instruction is “Save the file in your `site-config` directory.” Suggestions for the organization, however, can be provided.

By Function

You can organize the files according to the functional area of the deployment that they are associated with. When you are working with the files, it should be clear from this guide or a README what aspect of the deployment they relate to. Here is an example:

```
$deploy/  
├─ sas-bases/  
└─ site-config/
```

```

├─ authentication/
├─ cpu/
└─ storage/

```

By Type

Your deployment is based on a file that you create named `kustomization.yaml`, which contains five main blocks. When you modify an example file or use an overlay file, you will also be instructed to add a reference to that file in the `kustomization.yaml` file. Therefore, you can consider organizing the `site-config` directory according to the blocks in `kustomization.yaml`. Here is an example:

```

$deploy/
├─ sas-bases/
└─ site-config/
    ├─ configMapGenerator/
    ├─ patches/
    ├─ resources/
    ├─ secretGenerator/
    └─ transformers/

```

Directories for the SAS Viya Deployment Operator

Note: If you are deploying your SAS Viya software manually, skip this section.

Deploying with the SAS Viya Deployment Operator requires a more extensive directory structure that is based on the structure for the `$deploy` directory that is described in a preceding section.

```

├─ $license/
├─ $operator-deploy/
│   ├─ operator-base/
│   ├─ sas-bases/
│   └─ site-config/
│       ├─ cluster-wide-transformer.yaml
│       └─ transformer.yaml
└─ $deploy/
    ├─ sas-bases/
    └─ site-config/

```

The `$license` directory contains the `.jwt` file that licenses your software and the `*-certs.zip` file that contains the entitlement and CA certificates. If you have more than one deployment of SAS Viya, there will be multiple license and `*-certs.zip` files.

The `$operator-deploy/sas-bases` and `$operator-deploy/site-config` subdirectories are used only for configuring and deploying the SAS Viya Deployment Operator.

The `$deploy` directory is organized as described in a preceding section. Configurations and customizations for your SAS Viya deployment are made in the `$deploy/sas-bases` and `$deploy/site-config` subdirectories. You should have a uniquely named `$deploy` directory for each SAS Viya deployment.

Installation

<i>Initial kustomization.yaml File</i>	26
Overview	26
Create the File	27
<i>Common Customizations</i>	29
Overview	29
Using a Mirror Registry	30
Add the Update Checker to Your Deployment	32
Add a sitedefault File to Your Deployment	33
Configure TLS	34
Configure Container Security Settings	35
Enable Multi-tenancy	36
Configure SAS Image Staging	36
Configure High Availability	36
Configure PostgreSQL	37
Configure Open Distro for Elasticsearch	39
Specify SMP or MPP CAS	42
Configure CAS Settings	42
Configure SAS/CONNECT Settings	57
Configure External Storage Class for SAS Programming Run-Time Environment	60
Configure GPUs	60
Set Default SAS LOCALE and ENCODING in SAS Launcher Service	61
Change the Location of the NFS Server	61
Enable Access Methods Through LOCKDOWN System Option	61
Configure SSSD	62
Specify PersistentVolumeClaims to Use ReadWriteMany StorageClass	63
Configure SAS/ACCESS	64
<i>Create the SASDeployment Custom Resource</i>	64
Install the Orchestration Tool	64
Add an imagePullSecret for the SAS Viya Namespace	65
Run the create sas-deployment-cr Command	65
<i>Deploy the Software</i>	69
Deployment Using the SAS Viya Deployment Operator	69
Manual Deployment	71
<i>Readiness Service</i>	73
<i>Sign In as the sasboot User</i>	74

Initial kustomization.yaml File

Overview

The base kustomization.yaml file (`$deploy/kustomization.yaml`) is where you customize your Kubernetes deployment and allocate resources. The manifest that is used to deploy the software is based on the declarations in the kustomization.yaml file.

Note: Do not rename the kustomization.yaml file. The name of the file must be kustomization.yaml so that it is recognized by the Kustomize tool.

In this section, you are instructed to create a base kustomization.yaml file that deploys SAS with CAS in SMP mode and with an internal instance of PostgreSQL. It is intended to deploy an initial basic SAS Viya to help you understand how the base kustomization.yaml file works before adding more customizations. After you have deployed using the initial kustomization.yaml file, you can update it with more customizations for a new deployment or to change your initial deployment, or you can create your own kustomization.yaml file from scratch.

“[Common Customizations](#)” on page 29 contains the most commonly used configurations for deployments. To see the full list of configurations that are available for your order, go to `/$deploy/sas-bases/README.md` for README files in Markdown language or `$deploy/sas-bases/docs/index.htm` for README files in HTML.

IMPORTANT Before using an existing base kustomization.yaml file for deployments of subsequent versions of SAS Viya, review the deployment notes for changes that could invalidate the continued use of that file for the current version of the software. Although the deployment notes focus on updates, the same issues apply to a base kustomization.yaml file (and other .yaml files) that was created for the previous version of the software but are being used for the current version of the software.

Note: The README files that are written in the Markdown language can be read as plain text, but they are more useful if they are rendered as Markdown. SAS recommends that you use a tool that renders Markdown when using the README files.

Create the File

- 1 Change to the directory where you saved the deployment assets in [“Retrieve Required Files”](#) on page 13:

```
cd /$deploy
```

- 2 Create a kustomization.yaml file. Copy the following code example as the basis of the kustomization.yaml file. For each variable that is enclosed in braces ({}), replace the entire variable (including braces, text, and spaces) with the value that is appropriate for your deployment. For example, {{ NAME-OF-NAMESPACE }} should be replaced with a namespace name, such as `deploy`.

```
namespace: {{ NAME-OF-NAMESPACE }} 1
resources:
- sas-bases/base
- sas-bases/overlays/cert-manager-issuer 2
- sas-bases/overlays/network/networking.k8s.io 3
- sas-bases/overlays/cas-server
- sas-bases/overlays/internal-postgres
# If your deployment contains programming-only offerings only, comment out the next line
- sas-bases/overlays/internal-elasticsearch
- sas-bases/overlays/update-checker
- sas-bases/overlays/cas-server/auto-resources 4
configurations:
- sas-bases/overlays/required/kustomizeconfig.yaml
transformers:
# If your deployment does not support privileged containers or if your deployment
# contains programming-only offerings, comment out the next line 5
- sas-bases/overlays/internal-elasticsearch/sysctl-transformer.yaml
- sas-bases/overlays/required/transformers.yaml
- site-config/security/cert-manager-provided-ingress-certificate.yaml 2
- sas-bases/overlays/cas-server/auto-resources/remove-resources.yaml 4
# If your deployment contains programming-only offerings only, comment out the next line
- sas-bases/overlays/internal-elasticsearch/internal-elasticsearch-transformer.yaml
# Mount information
# - site-config/{{ DIRECTORY-PATH }}/cas-add-host-mount.yaml
components:
- sas-bases/components/security/core/base/full-stack-tls 6
- sas-bases/components/security/network/networking.k8s.io/ingress/
  nginx.ingress.kubernetes.io/full-stack-tls 6
patches:
- path: site-config/storageclass.yaml 7
  target:
    kind: PersistentVolumeClaim
    annotationSelector: sas.com/component-name in (sas-backup-job,sas-data-quality-
services,sas-commonfiles,sas-cas-operator,sas-pyconfig)
# License information
# secretGenerator:
# - name: sas-license
#   type: sas.com/license
#   behavior: merge
#   files:
# - SAS_LICENSE=license.jwt
```

```

configMapGenerator:
- name: ingress-input
  behavior: merge
  literals:
  - INGRESS_HOST={{ NAME-OF-INGRESS-HOST }}
- name: sas-shared-config
  behavior: merge
  literals:
  - SAS_SERVICES_URL=https://{{ NAME-OF-INGRESS-HOST }}:{{ PORT }} 8
# - SAS_URL_EXTERNAL_VIYA={{ EXTERNAL-PROXY-URL }} 9

```

- 1 If you are deploying with the SAS Viya Deployment Operator, the name of the namespace is not required.
- 2 These lines are appropriate for deployments using cert-manager for the certificate generator. For information about using OpenSSL instead of cert-manager, see the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).
- 3 If you are deploying on Red Hat OpenShift, replace this line with `- sas-bases/overlays/network/route.openshift.io`.
- 4 The nodes in the CAS node pool require taints in order to prevent Kubernetes from scheduling non-CAS workloads on them. If you enable the CAS auto-resources option, your designated CAS nodes must already be labeled. For more information, see [“Plan the Workload Placement” on page 1](#). If the nodes are not labeled, you must disable auto-resources and use hardcoded default resource settings. To disable this option, remove the following line from the resources section:

```
- sas-bases/overlays/cas-server/auto-resources
```

And remove the following line from the transformers section:

```
- sas-bases/overlays/cas-server/auto-resources/remove-resources.yaml
```

- 5 For more information about programming-only offerings, see [“Software Offerings” in *Getting Started with SAS Viya Operations*](#).
- 6 These lines are appropriate for NGINX Ingress on a full-stack TLS deployment using cert-manager as the certificate generator. For the other options available for these lines, see [“Configure TLS” on page 34](#).
- 7 For information about the storageclass.yaml file, see [“Specify PersistentVolumeClaims to Use ReadWriteMany StorageClass” on page 63](#).
- 8 The `SAS_SERVICES_URL` variable is used to provide the value for the `SERVICESBASEURL` option in compute server sessions. It should point to the host name and port of the cluster's Ingress for this deployment. The Ingress must be reachable from pods running inside the cluster. The `{{ NAME-OF-INGRESS-HOST }}` must match the value of the `INGRESS_HOST` variable.

If you are working with external reverse proxies or application gateways, such as Azure Application Gateway, `SAS_SERVICES_URL` and `INGRESS_HOST` can be set to a load balancer. The load balancer can be for the Ingress or external.

- 9 The `SAS_URL_EXTERNAL_VIYA` variable is optional. Use it to define an external proxy that should be used for access from outside your internal network (for example, in generated email messages or text messages). If you choose to

use the option, you must uncomment the line by removing the number sign (#).

- 3 Because this kustomization.yaml deploys Full-stack TLS using cert-manager as the certificate generator, follow the steps at [“Configure TLS” on page 34](#) for certificates for Ingress provided by cert-manager.
- 4 Because this kustomization.yaml deploys an internal instance of PostgreSQL, follow the steps for internal instances at [“Configure PostgreSQL” on page 37](#) to modify the appropriate .yaml file.
- 5 If you want to make changes to your configuration, see [“Common Customizations” on page 29](#), which is a description of the most frequently used configuration options. To see a list of all the configuration options for your software, go to `/deploy/sas-bases/README.md` for Markdown versions of the README files or `/deploy/sas-bases/docs/index.htm` for HTML versions of the README files.

Note: The README files written in the Markdown language can be read as plain text, but they are more useful if they are rendered as Markdown. SAS recommends that you use a tool that renders Markdown when using the README files.

- 6 To deploy your software:
 - If you are deploying your software with the SAS Viya Deployment Operator, continue to [“Create the SASDeployment Custom Resource” on page 64](#).
 - If you are deploying your software manually, go to [“Deploy the Software” on page 69](#).

Common Customizations

Overview

Configuring your SAS Viya deployment, including setting environment variables, is performed by modifying example and overlay files in your deployment assets and then adding references to those files to the base kustomization.yaml file. This section describes the most common deployment-wide configuration decisions that need to be made for the deployment and how to enact them.

IMPORTANT Product-specific configuration tasks are described in the README files included in your deployment assets. To see the full list of configurations that are available for your order, including the ones described in this section, go to `/deploy/sas-bases/README.md` for README files in Markdown language or `/deploy/sas-bases/docs/index.htm` for README files in HTML.

IMPORTANT Although the steps to configure the deployment are based on Kubernetes tasks and the Kustomize tool, the decisions behind them are based on how you want SAS Viya to run. Successfully configuring and deploying your software to run with the features you want requires conversation between the Kubernetes administrator and the SAS administrator. The Kubernetes administrator for your site should inform the SAS administrator of the options available for the deployment based on the README files in the deployment assets, perhaps even sharing the README files and their descriptions of the options. The Kubernetes administrator should then use the instructions in the README files to enable the options that the SAS administrator wants to use.

For more information about example and overlay files, see “[\\$deploy/sas-bases Directory](#)” on page 22.

Using a Mirror Registry

Note: For introductory information about mirror registries, including how to create them, see “[Create a Mirror Registry](#)” on page 107.

Add a Mirror Registry to Your SAS Viya Deployment

Note: Complete this task only if you are manually deploying your SAS Viya software. If you are deploying with the SAS Viya Deployment Operator, you have already completed this task while deploying the operator.

If you wish to use a mirror registry in your deployment, you must add a customization that reflects this. Perform the following steps:

- 1 Copy the `mirror.yaml` file from `sas-bases/examples/mirror/` to the site-config directory.
- 2 Open the `mirror.yaml` file and replace each instance of `{{ MIRROR-HOST }}` with the fully qualified domain name (FQDN) of the mirror registry and the registry namespace in this format:

FQDN/registry-namespace

For example:

`example.company.com/SASdeploy`

After you have made all the replacements, save and close the file.

- 3 In the base `kustomization.yaml` file, add the path to the `mirror.yaml` file to the transformers section, immediately after the entry for `sas-bases/overlays/`

`required/transformers.yaml`. For example, if you copied the `mirror.yaml` file to the top level of the `site-config` directory, the entry would look like this:

```
transformers:
...
- sas-bases/overlays/required/transformers.yaml
- site-config/mirror.yaml
```

- 4 If you are using an internal instance of PostgreSQL, add the following content to the `configMapGenerator` block in the base `kustomization.yaml` file:

```
configMapGenerator:
- name: input
  behavior: merge
  literals:
- IMAGE_REGISTRY={{ REGISTRY-HOST }}:{{ REGISTRY-PORT }}/{{ REGISTRY-
NAMESPACE }}
```

In the value for `IMAGE_REGISTRY`:

- `{{ REGISTRY-HOST }}` is required.
- `{{ REGISTRY-PORT }}` is optional. If `{{ REGISTRY-PORT }}` is omitted, the colon (:) that precedes it should be omitted also.
- `{{ REGISTRY-NAMESPACE }}` is optional. `{{ REGISTRY-NAMESPACE }}` is not the same as the Kubernetes namespace. If `{{ REGISTRY-NAMESPACE }}` is omitted, the slash (/) that precedes it should be omitted also.

Use ImagePullSecrets to Access the Mirror Registry

Note: If you are deploying your software with the SAS Viya Deployment Operator, skip this section.

If the target registry that you deposit the images in requires `imagePullSecrets` to access them, perform the following steps:

- 1 Create a representation of an `imagePullSecret` with `kubect`'s `--dry-run` option. For the steps to do so, see [Pull an Image from a Private Registry](#).
- 2 Add the following content to the base `kustomization.yaml` file in the `secretGenerator` block:

```
secretGenerator:
- name: sas-image-pull-secrets
  behavior: replace
  type: kubernetes.io/dockerconfigjson
  literals:
- '.dockerconfigjson={{ MY-IMAGEPULLSECRET }}'
```

Replace `{{ MY-IMAGEPULLSECRET }}` with the unencoded value of the `.dockerconfigjson` field of `imagePullSecret` you created in step 1.

Here is the command to obtain the value required for `{{ MY-IMAGEPULLSECRET }}`. The credentials used in these examples vary by cloud provider and by local cloud provider conventions adopted by your organization.

```
kubectl create secret docker-registry --dry-run=client regcred \
  --docker-server=cloud-provider-specific-registry-location \
  --docker-username=Docker-username \
  --docker-password=Docker-password \
  --docker-email=youremail@example.com \
  --output=jsonpath={.data\.dockerconfigjson} | base64 --decode
```

You should be familiar with the *cloud-provider-specific-registry-location* because you used it to create your mirror registry. For example, if you are using Microsoft Azure as your cloud provider, *cloud-provider-specific-registry-location* is `myregistry.azurecr.io`.

You can also optionally save `.dockerconfigjson` to a local file and reference it in your `secretGenerator` with the “files” keyword instead of “literals”.

Add the Update Checker to Your Deployment

Overview

The Update Checker cron job builds a report comparing the currently deployed release with available releases in the upstream repository. The report is written to the stdout of the launched job pod and indicates when new content related to the deployment is available.

If you use the initial `kustomization.yaml` file from this guide, the Update Checker is deployed by default. If you do not want to deploy the Update Checker, remove the `- sas-bases/overlays/update-checker` line from the resources block of the initial `kustomization.yaml` file.

For more information about using the Update Checker, see [“View the Update Checker Report” in SAS Viya Operations: Updating Software](#),

(Optional) Define Proxy Environment Variables

The Update Checker cron job is defined in the `sas-bases/overlays/update-checker/cronjobs.yaml` file. If you are using a proxy server for your SAS Viya deployment, these environment variables must be defined on the Update Checker cron job:

- `HTTP_PROXY`
- `HTTPS_PROXY`
- `NO_PROXY`

- 1 Copy `$deploy/sas-bases/examples/update-checker/proxy_transformer.yaml` file to `$deploy/site-config/update-checker/proxy_transformer.yaml`.
- 2 In the copied file, replace the value for `HTTP_PROXY` with the fully qualified domain name and port for the HTTP proxy. Replace the value for `HTTPS_PROXY` with the fully qualified domain name and port for the HTTPS proxy. Replace the value for `NO_PROXY` with a comma-separated list of fully qualified domain names that the proxy server should ignore.
- 3 Save and close `proxy_transformer.yaml`.
- 4 In the transformers section of the base `kustomization.yaml` file, add a reference to the `proxy_transformer.yaml` file:

```
transformers:
...
- site-config/update-checker/proxy_transformer.yaml
```

Add a sitedefault File to Your Deployment

Note: The extension for the `sitedefault` file for SAS Viya 4 is different from the file extension used in SAS Viya 3.x.

- SAS Viya 3.x uses `sitedefault.yml`.
- SAS Viya 4 uses `sitedefault.yaml`.

Instructions in this section include references to both `sitedefault` files, depending on which version of SAS Viya is being referred to. Take note of the file extension when it is used.

The `sitedefault.yaml` file, which is located in the `$deploy/sas-bases/examples/configuration` directory, is used primarily for the bulk loading of configuration values. After the initial deployment, the `sitedefault.yaml` file cannot be used to modify an existing value and to deploy the software again. The `sitedefault.yaml` file can be used only to set new property values that have not already been set.

Therefore, SAS recommends that you do **not** use the `sitedefault.yaml` file for the initial deployment of your SAS Viya software, except where specifically described in this guide. You can, however, use a `sitedefault` file from a previous SAS Viya release.

To load a SAS Viya 3.x `sitedefault.yml` file in your deployment:

- 1 Ensure that the `sitedefault.yml` file is in the `site-config` directory.
- 2 Add the following content to the `kustomization.yaml` file in the `secretGenerator` block:

```
- name: sas-consul-config
  behavior: merge
  files:
    - SITEDEFAULT_CONF={{ DIRECTORY-PATH }}/sitedefault.yml
```

Configure TLS

Follow these steps to deploy using cert-manager as the certificate generator.

Note: The example kustomization.yaml file, located at [“Create the File” on page 27](#), represents a deployment with NGINX TLS for full-stack TLS mode using cert-manager as the certificate generator.

- 1 Copy the `$deploy/sas-bases/examples/security/cert-manager-provided-ingress-certificate.yaml` file to `$deploy/site-config/security/cert-manager-provided-ingress-certificate.yaml`.
- 2 Open the file and replace `{{ CERT-MANAGER-ISSUER-NAME }}` with the name of the cert-manager issuer that will issue certificates used to make TLS connections to the SAS Viya applications. For example, `sas-viya-issuer`.
- 3 In the transformers block of the base kustomization.yaml, add a reference to the `cert-manager-provided-ingress-certificate.yaml` file:

```
transformers:
...
- site-config/security/cert-manager-provided-ingress-certificate.yaml
...
```

- 4 Add the following lines to the components block of the base kustomization.yaml file. Create the components block if it does not already exist.

- a Add a reference to the appropriate subdirectory in `$deploy/sas-bases/components/security/core/base/`:

```
components:
- sas-bases/components/security/core/base/TLS-mode
```

Replace *TLS-mode* with one of the following based on the TLS mode you are deploying:

- `front-door-tls`
- `full-stack-tls`
- `truststores-only`

Note: If you use `truststores-only` as the *TLS-mode*, skip part b of this step.

- b Add a reference to the appropriate subdirectory in `$deploy/sas-bases/components/security/network/`:

```
components:
- sas-bases/components/security/network/ingress-technology/TLS-mode
```

Replace *ingress-technology* with the technology that handles traffic entering your cluster:

- `networking.k8s.io/ingress/nginx.ingress.kubernetes.io`

- `route.openshift.io/route`

Replace *TLS-mode* with one of the following based on the TLS mode you are deploying:

- `front-door-tls`
- `full-stack-tls`

Note: For information about the required additions to the base `kustomization.yaml` for using OpenSSL as the certificate generator, see the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

SAS strongly recommends that you safeguard your passwords and data by securing network communication. You can choose not to use TLS, but communication between the pods in your deployment will be unsecured. If you accept that risk or want to conduct experiments using fake data and credentials, you can eliminate network security by deleting the following lines from the example `kustomization.yaml`:

- From the transformers block:
 - `- site-config/security/cert-manager-provided-ingress-certificate.yaml`
- From the components block:
 - `- sas-bases/components/security/core/base/full-stack-tls`
 - `- sas-bases/components/security/network/networking.k8s.io/ingress/nginx.ingress.kubernetes.io/full-stack-tls`

For more information about TLS and your SAS Viya deployment, see the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

Configure Container Security Settings

Note: If you are deploying on Red Hat OpenShift and have completed the steps in “[Additional Security](#)” on page 21, you have performed the necessary steps and should skip this section.

You can change the default container security settings in a SAS Viya deployment, such as removing, adding, or updating settings in the podSpecs. SAS has provided example and overlay files to manage the `fsGroup` field and the secure computing mode (seccomp) profile for your deployment.

- The `fsGroup` field defines a special supplemental group that assigns a group ID (GID) for all containers in the pod.
- Seccomp is a Linux kernel feature used to restrict actions available within a container.

There are many reasons why an administrator might want to modify these settings. However, if you are deploying on Red Hat OpenShift, they must be modified in order to take advantage of OpenShift's built-in security context constraints. For more information about these settings, see the README file at `$deploy/sas-bases/examples/security/container-security/README.md` (for Markdown format) or `$deploy/sas-bases/docs/modify_container_security_settings.htm` (for HTML format).

Enable Multi-tenancy

Note: Multi-tenancy is not supported on Red Hat OpenShift at this time.

By default, your SAS Viya deployment is not multi-tenant. To make your SAS Viya deployment multi-tenant, follow the instructions in the README file at `$deploy/sas-bases/examples/multi-tenant/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/multi-tenant_deployment.htm` (for HTML format).

Note: The decision to enable multi-tenancy must be made before deployment. You cannot change the multi-tenancy status of your deployment after the software has been deployed. The only way to change the status of multi-tenancy in a deployment is to re-deploy the software.

Configure SAS Image Staging

By default, SAS Image Staging starts pods on nodes via a daemonset at approximate two-minute intervals to ensure that relevant images have been pulled to hosts. While this behavior accomplishes the goal of pulling images to nodes and decreasing start-up times, some users may want more intelligent and specific control with less churn in Kubernetes. To accomplish these goals, configure SAS Image Staging to take advantage of a node list to further decrease start-up times and target specific nodes for pulling.

For information about both methods of using SAS Image Staging, including a comparison of their relative advantages and disadvantages, see the README file at `$deploy/sas-bases/examples/sas-prepull/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/sas_image_staging_configuration_option.htm` (for HTML format).

Configure High Availability

SAS Viya can be deployed as a High Availability (HA) system. In this mode, SAS Viya has redundant stateless and stateful services to handle service outages, such as an errant Kubernetes node. A Kustomize transformer enables HA in SAS Viya among the stateless microservices. Stateful services, with the exception of SMP and Open Distro for Elasticsearch, are enabled as HA at initial deployment.

To enable HA, add a reference to the `enable-ha-transformer.yaml` file to the base `kustomization.yaml` file:

```
...
transformers:
...
- sas-bases/overlays/scaling/ha/enable-ha-transformer.yaml
...
```

Note: To enable HA for Open Distro for Elasticsearch, see the README file located at `$deploy/sas-bases/examples/configure-elasticsearch/internal/topology/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_a_default_topology_for_open_distro_for_elasticsearch.htm` (for HTML format).

For more information about this transformer file and disabling HA, see the README file at `$deploy/sas-bases/examples/scaling/ha/README.md` (for Markdown) or at `$deploy/examples/docs/high_availability_ha_in_sas_viya.htm` (for HTML).

Note: The instructions in this section increase the number of replicas of your SAS Viya deployment, making it more resilient to pod or node failure and increasing its availability. However, your SAS Viya environment probably has dependencies on software that is running in other namespaces in the same cluster. For example, software like `cert-manager`, `NGINX Ingress controller`, and the SAS Viya Monitoring solution for Kubernetes might be critical to the availability of SAS Viya and may have been deployed by default with unique replicas, making them less highly available than SAS Viya itself. If you want to increase the availability of other software, consult the documentation for that software for more information.

Furthermore, an over-tainting of the nodes for SAS Viya can result in third-party software being locked out of the cluster in spite of available spare capacity. In order to achieve maximum overall availability, either dedicate some nodes to this software or add tolerations to it so it can more easily run on the same nodes as SAS Viya.

Configure PostgreSQL

Based on your decision about your PostgreSQL instance (see [“Internal versus External PostgreSQL Instances”](#) in *System Requirements for SAS Viya*), you must perform steps to deploy PostgreSQL (internal) or connect to an existing PostgreSQL instance (external).

External Instance of PostgreSQL

To set up an external instance of PostgreSQL:

- 1 For each instance of external PostgreSQL in your SAS Viya deployment, make a copy of the `$deploy/examples/postgres/configure/external-postgres.yaml` and place it in your `site-config` directory. Each copy should be uniquely named.

- 2 Using the comments in the file, revise each copy of `external-postgres.yaml` with the appropriate values for the external instance of PostgreSQL that it represents.
- 3 In the resources block of the base `kustomization.yaml`, add a reference for each `external-postgres.yaml`. Here is an example with two external instances of PostgreSQL:

```
resources:
...
- site-config/postgres/configure/external-postgres-1.yaml
- site-config/postgres/configure/external-postgres-2.yaml
...
```

- 4 In the transformers block of the base `kustomization.yaml` file, add a reference to the `external-postgres-transformer.yaml` overlay:

```
transformers:
...
- sas-bases/overlays/external-postgres/external-postgres-transformer.yaml
```

- 5 If you use the initial `kustomization.yaml` file located at [“Create the File” on page 27](#), remove the `- sas-bases/overlays/internal-postgres` line from the resources block of that file.

For more information, see the “Configure PostgreSQL” README file at `$deploy/sas-bases/examples/postgres/configure/README.md` (for Markdown format) or `$deploy/sas-bases/docs/configure_postgresql.htm` (for HTML format),

Internal Instance of PostgreSQL

If you are using an internal instance of PostgreSQL, add a reference to the internal PostgreSQL content to the resources block of the base `kustomization.yaml` file. Here is an example:

```
resources:
...
- sas-bases/overlays/internal-postgres
...
```

The `kustomization.yaml` file from [“Initial kustomization.yaml File” on page 26](#) includes this reference. For additional information about configuring an internal instance of PostgreSQL, see the “Configuration Settings for PostgreSQL Database Cluster” README located at `$deploy/sas-bases/examples/postgres/custom-config/README.md` (for Markdown format) or `$deploy/sas-bases/docs/configuration_settings_for_postgresql_database_cluster.htm` (for HTML format).

Note: If you choose an internal instance of PostgreSQL, [Appendix 3, “PostgreSQL and Crunchy Data,” on page 129](#) contains a list of all the containers that are available, including some that are not deployed by default. You should review the list to determine if there is functionality you want to add to your PostgreSQL instance.

Configure CDS PostgreSQL

Note: For more information about CDS PostgreSQL, see [“Common Data Store” in SAS Viya: Configuration Reference](#).

Several SAS Viya offerings include a second instance of PostgreSQL referred to as CDS PostgreSQL. The CDS PostgreSQL instance is used because the character of the data used by those offerings is hierarchically different than the data generally stored in the primary PostgreSQL database. The separation into two different databases allows them to be tuned individually, in turn enhancing the performance of both.

If your software order includes at least one of these offerings, CDS PostgreSQL must be customized as well. To determine if your software includes CDS PostgreSQL, look for the `$deploy/sas-bases/overlays/sas-planning` directory. If the directory exists, then continue with the tasks in this section. If that directory does not exist, you should skip the rest of this section.

If you want to use an internal instance of CDS PostgreSQL, follow the steps in the “Configure an Internal PostgreSQL Instance for sas-planning” section of the SAS Planning README file, located at `$deploy/sas-bases/overlays/sas-planning/README.md` (for Markdown format) or `$deploy/sas-bases/docs/configuring_customizations_for_sas-planning.htm` (for HTML format).

If you want to use an external instance of CDS PostgreSQL, follow the steps in the “Configure an External PostgreSQL Instance for sas-planning” section of the SAS Planning README file, located at `$deploy/sas-bases/overlays/sas-planning/README.md` (for Markdown format) or `$deploy/sas-bases/docs/configuring_customizations_for_sas-planning.htm` (for HTML format). Additionally, if you use the example `kustomization.yaml` file from [“Initial kustomization.yaml File” on page 26](#), remove the `- sas-bases/overlays/internal-postgres` line from the resources block.

Configure Open Distro for Elasticsearch

Note: Offerings that are designated programming-only do not include Elasticsearch. If your SAS Viya order contains programming-only offerings, skip this section. For more information about programming-only offerings, see [“Software Offerings” in Getting Started with SAS Viya Operations](#).

Initial Customizations

[Open Distro for Elasticsearch](#) is an Apache 2.0-licensed distribution of [Elasticsearch](#) enhanced with enterprise security. SAS Viya includes Open Distro for Elasticsearch and uses its distributed search cluster in infrastructure and solution services. Some

additions to the base kustomization.yaml file must be made to configure Open Distro for Elasticsearch.

Note: The example kustomization.yaml file, located at “[Initial kustomization.yaml File](#)” on page 26, includes these customizations.

- 1 Add the following line to the resources block of the base kustomization.yaml file:

```
resources:
...
- sas-bases/overlays/internal-elasticsearch
...
```

- 2 Add the following line to the transformers block of the base kustomization.yaml file:

```
transformers:
...
- sas-bases/overlays/internal-elasticsearch/internal-elasticsearch-transformer.yaml
...
```

Configure Default Virtual Memory Resources

Note: If you are deploying on Red Hat OpenShift and have completed the steps in “[Security Context Constraints and Service Accounts](#)” on page 15, you have performed the necessary steps and should skip this section.

The Open Distro for Elasticsearch Pods require additional virtual memory resources. In order to provide these memory resources, a transformer uses a privileged container to set the virtual memory for the mmapfs directory to the required level. Therefore, privileged containers must be permitted by your Pod security policies. For more information about Pod security policies, see <https://kubernetes.io/docs/concepts/policy/pod-security-policy/>.

You have three options:

- If privileged containers are enabled, add a reference to the sysctl-transformer.yaml file to the transformers block of the base kustomization.yaml file. This transformer must be included after any TLS transformers and before the sas-bases/overlays/required/transformers.yaml transformer.

Note: The sysctl-transformer.yaml transformer uses a privileged container to set [vm.max_map_count](#). If privileged containers are not allowed in your deployment, do not add this line.

Here is an example:

```
transformers:
...
- sas-bases/overlays/network/ingress/security/transformers/...
- sas-bases/overlays/internal-elasticsearch/sysctl-transformer.yaml
```

```
- sas-bases/overlays/required/transformers.yaml
...
```

Note: Using this option requires modifying the OpenShift SCC for sas-opendistro to allow it. For more information, see the README file at `$deploy/sas-bases/examples/configure-elasticsearch/internal/openshift/README.md` (for Markdown format) or `$deploy/sas-bases/docs/open_distro_for_elasticsearch_on_red_hat_openshift.htm` (for HTML format).

- If privileged containers are not allowed in your environment, a Kubernetes administrator with elevated permissions can set the virtual memory manually before performing the SAS Viya deployment. All nodes that run workloads in the [stateful workload class](#) are affected by this requirement.

To configure the virtual memory settings for mmapfs manually:

- 1 Log on to the first stateful node as root or with a sudoers account.
- 2 Set the virtual memory using the appropriate method:
 - To set the value permanently, use your preferred text editor to modify `/etc/sysctl.conf` or the equivalent in your environment. Update the `vm.max_map_count` setting to `262144` and save the file.
 - To set the value temporarily, run the following command:

```
sysctl -w vm.max_map_count=262144
```

- 3 (Optional) Verify the modified setting:

```
sysctl vm.max_map_count
```

- 4 Repeat the previous steps on each node that is labeled for stateful workloads.

If you are using a managed Kubernetes cluster, your cloud provider probably provisions the nodes dynamically. In this instance, be aware that manual modifications do not persist after a restart of a Kubernetes node. The cluster administrator must use an alternative method to save the `vm.max_map_count` setting.

- You can disable the use of mmap at a cost of performance and memory usage. To disable mmap, include a reference to the `disable-mmap-transformer.yaml` overlay in the transformers block of the base `kustomization.yaml` file.

```
transformers:
...
- sas-bases/overlays/internal-elasticsearch/disable-mmap-transformer.yaml
```

Configure a StorageClass

Deploying Open Distro for Elasticsearch requires a StorageClass that provides block storage (such as virtual disks) or a local file system mount to store the search indices. For the instructions to configure such a StorageClass for all cloud providers, see the “Configure a Default StorageClass for Open Distro for Elasticsearch” README, located at `$deploy/sas-bases//examples/configure-elasticsearch/`

`internal/storage/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_a_default_storageclass_for_open_distro_for_elasticsearch.htm` (for HTML format).

Configure High Availability

To enable HA for Open Distro for Elasticsearch, see the README file located at `$deploy/sas-bases/examples/configure-elasticsearch/internal/topology/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_a_default_topology_for_open_distro_for_elasticsearch.htm` (for HTML format).

Specify SMP or MPP CAS

Your deployment of SAS Cloud Analytic Services (CAS) can be performed on a single node (SMP) or across several nodes (MPP). The base `kustomization.yaml` file from “[Initial kustomization.yaml File](#)” on page 26 includes the reference that is required for deploying SMP or MPP CAS. If you do not make any changes to the files in `$deploy/sas-bases/overlays/cas-server`, CAS is deployed as SMP.

To deploy MPP CAS, follow the instructions in the README file at `$deploy/sas-bases/overlays/cas-server/README.md` (for Markdown) or `$deploy/sas-bases/docs/cas_server_for_sas_viya.htm` (for HTML) to modify the appropriate files.

Note: Deployments that enable multi-tenancy should use SMP CAS. When additional tenants are onboarded, the decision whether to use SMP or MPP CAS should be made for each tenant.

For more information about SMP and MPP deployments of CAS, see “[What is the CAS Server, SMP, and MPP?](#)” in *Getting Started with SAS Viya Operations*.

Configure CAS Settings

Mount hostPaths and Data Connectors for the CAS Server

Data storage in containers is ephemeral. When a container is deleted, the stored data is lost. For durable storage, data should be maintained in persistent volumes outside the Kubernetes cluster. Data remains intact, regardless of whether the containers that the storage is connected to are terminated.

To connect data storage outside the cluster to your SAS Viya deployment:

- 1 Copy `$deploy/sas-bases/examples/cas/configure/cas-add-host-mount.yaml` to your `/site-config` directory.

Note: For more information about the `/site-config` directory and its structure, see [“\\$deploy/site-config Directory” on page 23](#).

- 2 In the new `cas-add-host-mount.yaml` file, replace the variables with actual values. Variables are enclosed in braces (`{ }`) and spaces. When replacing a variable with an actual value, ensure that the braces, spaces, and the hyphenated variable name are removed.
- 3 Save and close the new `cas-add-host-mount.yaml` file.
- 4 In the base `kustomization.yaml` file, add the path to your new `cas-add-host-mount.yaml` file to the transformers block:

```
...
transformers:
- site-config/{{ DIRECTORY-PATH }}/cas-add-host-mount.yaml
...
```

- 5 Save and close the `kustomization.yaml` file.

Note: You need a version of the `cas-add-host-mount.yaml` file for each persistent volume that you want to use. Repeat the steps in this section for each persistent volume. Use a different name for the `.yaml` file each time. Try to use a name that indicates the purpose of the file.

Change accessMode

The default `accessMode` for the `cas-default-data` (referred to as “CASDATADIR” in SAS Viya 3.X) and `cas-default-permstore` persistentVolumeClaims is `ReadWriteMany`, because it is required for any backup controllers for CAS. It is not required for deployments with SMP CAS, but changing the access mode complicates a possible transition from SMP to MPP in the future.

To change the access mode for either `cas-default-data` or `cas-default-permstore`:

- 1 Copy `$deploy/sas-bases/examples/cas/configure/cas-storage-access-modes.yaml` to your `/site-config` directory.

Note: For more information about the `/site-config` directory and its structure, see [“\\$deploy/site-config Directory” on page 23](#).

- 2 In the new `cas-storage-access-modes.yaml` file, replace the variables with actual values. Variables are enclosed in braces (`{ }`) and spaces. To replace a variable with an actual value, ensure that the braces, spaces, and hyphenated variable name are removed.
- 3 Save and close the new `cas-storage-access-modes.yaml` file.

- 4 In the base kustomization.yaml file, add the path to your new cas-storage-access-modes.yaml file to the transformers block:

```
...
transformers:
- site-config/{{ DIRECTORY-PATH }}/cas-storage-access-modes.yaml
...
```

- 5 If you are using the initial kustomization.yaml file, go to the patches block. Remove `sas-cas-operator` from the parenthetical list in the `annotationSelector` value.

Note: For more information about the initial kustomization.yaml file, see [“Initial kustomization.yaml File” on page 26](#).

- 6 Save and close the kustomization.yaml file.

Adjust RAM and CPU Resources for CAS Servers

If you use the initial kustomization.yaml file, the CAS operator determines the amount of RAM required for your deployment based on available RAM on the Kubernetes nodes where CAS is running. When you set the RAM resources manually, you must also set the CPU resources manually.

Note: For auto-resourcing to work appropriately, you must have set labels on your node. See [“Plan the Workload Placement” on page 1](#) for more information.

If you prefer to set your own RAM and CPU resources, perform the following steps.

- 1 Copy `$deploy/sas-bases/examples/cas/configure/cas-manage-cpu-and-memory.yaml` to your `/site-config` directory.

Note: For more information about the `/site-config` directory and its structure, see [“\\$deploy/site-config Directory” on page 23](#).

- 2 In the new `cas-manage-cpu-and-memory.yaml` file, replace the variables with actual values. Variables are enclosed in braces (`{ }`) and spaces. To replace a variable, ensure that the braces, spaces, and hyphenated variable name are removed.
- 3 Save and close the new `cas-manage-cpu-and-memory.yaml` file.
- 4 In the base kustomization.yaml file, remove `- sas-bases/overlays/cas-server/auto-resources` from the resources block. Also remove `- sas-bases/overlays/cas-server/auto-resources/remove-resources.yaml` from the transformers block.
- 5 In the base kustomization.yaml file, add the path to your new `cas-manage-cpu-and-memory.yaml` file to the transformers block:


```
...
transformers:
- site-config/{{ DIRECTORY-PATH }}/cas-manage-cpu-and-memory.yaml
...
```

- 6 Save and close the kustomization.yaml file.

Change the Number of Workers for MPP CAS

Note: This customization can be performed only for deployments enabling MPP CAS.

By default, MPP CAS has two workers. Perform the following steps to change the number of workers before or after the initial deployment of your SAS Viya software.

Note: If you want to change the number of workers after the initial deployment of SAS Viya, adding workers and having them join the grid does not require a restart. However, existing SAS sessions will not reallocate or load balance to use the new workers. New sessions should take advantage of the new workers.

Removing workers after the initial deployment requires deleting the CAS deployment, modifying the YAML file, restarting the CAS server, reloading your data, and starting new SAS sessions.

- 1 Copy `$deploy/sas-bases/examples/cas/configure/cas-manage-workers.yaml` to your `/site-config` directory.

Note: For more information about the `/site-config` directory and its structure, see [“\\$deploy/site-config Directory” on page 23](#).

- 2 In the new `cas-manage-workers.yaml` file, replace the variables with actual values. Variables are enclosed in braces (`{ }`) and spaces. To replace a variable with a value, ensure that the braces, spaces, and hyphenated variable name are removed.
- 3 Save and close the new `cas-manage-workers.yaml` file.
- 4 In the base `kustomization.yaml` file, add the path to your new `cas-manage-workers.yaml` file to the `transformers` block:

```
...
transformers:
- site-config/{{ DIRECTORY-PATH }}/cas-manage-workers.yaml
...
```

- 5 Save and close the kustomization.yaml file.

Add a Backup Controller for MPP CAS

Note: This customization can be performed only for deployments with CAS in MPP mode.

- 1 Copy `$deploy/sas-bases/examples/cas/configure/cas-manage-backup.yaml` to your `/site-config` directory.

Note: For more information about the `/site-config` directory and its structure, see [“\\$deploy/site-config Directory” on page 23](#).

- 2 In the new `cas-manage-backup.yaml` file, replace the variable with the value 0 or 1. The value 0 indicates that you do not want a backup controller, and the value 1 indicates that you want a backup controller.
- 3 Save and close the new `cas-manage-backup.yaml` file.
- 4 In the base `kustomization.yaml` file, add the path to your new `cas-manage-backup.yaml` file to the transformers block:

```
...
transformers:
- site-config/{{ DIRECTORY-PATH }}/cas-manage-backup.yaml
...
```

- 5 Save and close the `kustomization.yaml` file.

Tune CAS_DISK_CACHE

About CAS_DISK_CACHE

The CAS server uses the directory or directories referred to as the CAS Disk Cache as a scratch area. It is associated with the environment variable `CASENV_CAS_DISK_CACHE` and has two primary purposes:

- 1 As data is loaded into memory, it is organized in blocks. Each time a block reaches the default block size of 16Mb, the block is copied to the CAS Disk Cache. The copied block can be re-read back into memory quickly if memory use becomes high and the original data must be freed from memory.
- 2 For a distributed CAS server (MPP), copies of the blocks are transferred to CAS worker pod for fault tolerance. Those copies are also stored in the CAS Disk Cache of the receiving CAS Worker.

A secondary use of the cache is for files that are uploaded to the server. By default, a copy of the file is temporarily stored on the CAS controller in its CAS Disk Cache.

To specify a different location, see [“Storage Location for Uploaded Files”](#) on page 50.

About the Default Configuration

By default, the server is configured to use a directory that is named `/cas/cache` on each controller and worker pod. This directory is provisioned as a Kubernetes `emptyDir` and uses disk space from the root volume of the Kubernetes node.

The default configuration is acceptable for testing and evaluation, but not for production workloads. If disk space in the root volume of the node becomes low, then Kubernetes begins evicting pods. The pod is unlikely to be rescheduled.

When the server stores a block in the cache, the server uses a configure technique that involves opening a file, deleting the file, and then holding the handle to the deleted file. The negative consequence to this technique is that Kubernetes cannot monitor the disk use in the cache.

Choose the Best Storage

The server uses memory mapped I/O for the blocks in the cache. The best performance is provided by using disks that are local to the node for each controller and worker pod. If possible, use disks that provide high data transfer rates such as NVMe or SSD.

If you follow the best practices for workload placement, then no other pods are scheduled on a node that is used by CAS. Even if the root volume is sufficiently large, it is likely that the performance yielded by the root volume will be lower than that of an Ephemeral drive, assuming one is available to the node.

A better strategy is to use a disk that is attached to the node. If the server fills the disk with blocks, the server logs an error rather than Kubernetes evicting the pod. An end user receives the following message when the server runs out of disk space used for the cache on any node.

```
Cloud Analytic Services failed writing to system disk space. Please contact your
administrator.
```

Note: The disk that is used does not need to persist beyond the duration of the pod and does not need to be backed up. Ephemeral storage is ideal.

Use a hostPath for CAS Disk Cache

Most cloud providers offer virtual machines that include a temporary disk for ephemeral storage. Typically, the disk is available at `/dev/sdb1` or a similarly named device. Some cloud providers automatically mount the device on the `/mnt` directory for the VM.

In order to leverage those alternate disks, you can use a Kubernetes `hostPath` instead of an `emptyDir`. The SAS Viya deployment requires that those temporary disks are already mounted and available on the CAS nodes and that the path is identical on all nodes.

Single Disk for CAS Disk Cache

- 1 In your `$deploy/site-config/` directory, create a file named `cas_disk_cache-config.yaml`.
- 2 Use the following content in the `cas_disk_cache-config.yaml` file. Replace the variables in the brackets, and the brackets themselves, with values that match your environment.

```
# # this defines the volume and volumemount for CAS DISK CACHE location
---
apiVersion: builtin
kind: PatchTransformer
metadata:
  name: cas-cache-hostpath
patch: |-
  - op: add
    path: /spec/controllerTemplate/spec/volumes/-
    value:
      name: cas-cache-nvme0
      hostPath: # # hostPath, is the path on the host, outside the pod
        path: {{/mnt-nvme0}}
  - op: add
    path: /spec/controllerTemplate/spec/containers/0/volumeMounts/-
    value:
      name: cas-cache-nvme0
      mountPath: /cas/cache-nvme0 # # mountPath is the path inside the
pod that CAS will reference
  - op: add
    path: /spec/controllerTemplate/spec/containers/0/env/-
    value:
      name: CASENV_CAS_DISK_CACHE
      value: "/cas/cache-nvme0" # # This has to match the value that
is inside the pod
target:
  version: vlalpha1
  group: viya.sas.com
  kind: CASDeployment
  # # Target filtering: chose/uncomment one of these option:
  # #   To target only the default CAS server (cas-shared-default) :
  labelSelector: "sas.com/cas-server-default"
  # #   To target only a single CAS server (e.g. MyCAS) other than
default:
  # name: {{MyCAS}}
  # #   To target all CAS Servers
  # name: .*
```

- 3 In the base `kustomization.yaml` file, add the path to your new `cas_disk_cache-config.yaml` file to the transformers block:

```
...
transformers:
...
- site-config/cas_disk_cache-config.yaml
...
```

Microsoft Azure and other cloud providers offer VMs with NVMe storage. Make sure the volume is formatted with an xfs or ext4 file system and is mounted by the VM.

Multiple Disks for CAS Disk Cache

If you use nodes with more than one high-performance disk, you can use more than one disk for the CAS Disk Cache. The server uses a round-robin algorithm for storing blocks on multiple disks.

- 1 In your `$deploy/site-config/` directory, create a file named `cas_disk_cache-config.yaml`.
- 2 Use the following content in the `cas_disk_cache-config.yaml` file. Replace the variables in the brackets, and the brackets themselves, with values that match your environment.

```
# # this defines the volume and volumemount for CAS DISK CACHE location
---
apiVersion: builtin
kind: PatchTransformer
metadata:
  name: cas-cache-hostpath
patch: |-
  - op: add
    path: /spec/controllerTemplate/spec/volumes/-
    value:
      name: cas-cache-nvme0
      hostPath: # # hostPath, is the path on the host, outside the pod
        path: {{/mnt-nvme0}}
  - op: add
    path: /spec/controllerTemplate/spec/volumes/-
    value:
      name: cas-cache-nvme1
      hostPath: # # hostPath, is the path on the host, outside the pod
        path: {{/mnt-nvme1}}
  - op: add
    path: /spec/controllerTemplate/spec/containers/0/volumeMounts/-
    value:
      name: cas-cache-nvme0
      mountPath: /cas/cache-nvme0 # # mountPath is the path inside the
pod that CAS will reference
  - op: add
    path: /spec/controllerTemplate/spec/containers/0/volumeMounts/-
    value:
      name: cas-cache-nvme1
      mountPath: /cas/cache-nvme1 # # mountPath is the path inside the
pod that CAS will reference
  - op: add
    path: /spec/controllerTemplate/spec/containers/0/env/-
    value:
      name: CASENV_CAS_DISK_CACHE
      value: "/cas/cache-nvme0:/cas/cache-nvme1" # # This has to match
the value that is inside the pod
target:
  version: v1alpha1
  group: viya.sas.com
```

```

kind: CASDeployment
# # Target filtering: chose/uncomment one of these option:
# #   To target only the default CAS server (cas-shared-default) :
labelSelector: "sas.com/cas-server-default"
# #   To target only a single CAS server (e.g. MyCAS) other than
default:
# name: {{MyCAS}}
# #   To target all CAS Servers
# name: .*

```

- 3 In the base kustomization.yaml file, add the path to your new cas_disk_cache-config.yaml file to the transformers block:

```

...
transformers:
...
- site-config/cas_disk_cache-config.yaml
...

```

The preceding sample suggests that two NVMe disks are mounted on the node at /mnt-nvme0 and /mnt-nvme1. Steps to perform that action are not shown in this documentation.

Configure Block Size

By default, the server uses a 16 MB block size. If the site accesses very large tables exclusively, you can configure a larger block size to reduce the chance of running out of file handles. Set the CASCFG_MAXTABLEMEM environment variable to the preferred value by adding the following block of code to the end of the patch block of your cas_disk_cache-config.yaml file.

```

- op: add
  path: /spec/controllerTemplate/spec/containers/0/env/-
  value:
    name: CASCFG_MAXTABLEMEM
    value: {{ BLOCKSIZE }}

```

The value for {{ BLOCKSIZE }} should be a numerical value followed by units (K=kilobytes, M=megabytes, or G=gigabytes). The default is 16M.

If a variety of table sizes is used, then individual users can set the MAXTABLEMEM session option on a case-by-case basis.

Storage Location for Uploaded Files

An upload is a data transfer of an entire file to the server, such as a SAS data set in SAS7BDAT format or a CSV file. The client, such as SAS, Python, or a web browser, performs no processing on the file. The server performs any processing that is needed, such as parsing records from a CSV file.

```

- op: add
  path: /spec/controllerTemplate/spec/containers/0/env/-
  value:
    name: CASENV_CAS_CONTROLLER_TEMP
    value: {{ MOUNT-PATH-TO-VOLUME }}

```

Ensure that the path you use for {{ MOUNT-PATH-TO-VOLUME }} is enclosed by double quotation marks, such as "/cas/cache-nvme0".

Configure External Access to CAS

Overview of CAS Connectivity

By default, a single CAS server is configured during the deployment process and is accessible to SAS services and web applications that are deployed in the Kubernetes cluster. For example, SAS Visual Analytics, SAS Studio, and other SAS software can work with CAS and do not require any additional configuration.

In addition, an HTTP Ingress is enabled that provides access to CAS from outside the cluster to clients that use REST. This Ingress can be used with clients such as Python SWAT.

The Ingress Controller that is configured for your cluster enables connectivity to CAS at an HTTP path like `https://www.example.com/cas-shared-default-http/`.

Note: Use the path as shown for clients such as Python SWAT. For curl, use a path such as `/cas-shared-default-http/cas`. This document shows the path that is appropriate for Python SWAT.

Note: The default instance of the CAS server is referenced in this example and the rest of this topic. If you add more than one server, then the Ingress or Service name uses the server instance name instead of the word “default”.

Optional Connectivity

There are two uses of CAS that require additional configuration:

- **Connections from SAS 9.4, SAS Viya 3.5, or other binary clients.** If you want to connect to CAS from SAS Viya 3.5, SAS 9.4, or use a binary connection with open programming clients such as Python, R, and Java, you can enable a binary connection.
- **Connections to CAS from SAS Data Connectors.** For information about enabling connectivity for SAS/ACCESS and data connectors, see the README file `$deploy/sas-bases/examples/data-access/README.md` (for Markdown) or `$deploy/sas-bases/docs/configuring_sasaccess_and_data_connectors_for_sas_viya_4.htm` (for HTML).

About Binary Connectivity

Most clients can use a binary connection to the CAS server. Typically, performance is better than HTTP because the data stream is more compact than REST.

If you want to connect from SAS Viya 3.5 or SAS 9.4, then you must enable binary communication. You can use the node port or load balancer as described here or

you can configure a custom Ingress to proxy TCP port 5570. Configuring a custom Ingress is not described in this documentation.

Optional Binary and HTTP Services

You can enable two services that provide external access to CAS for programmers. One service provides binary communication and the other service provides HTTP communication for REST. The HTTP service is an alternative to using the HTTP Ingress that is enabled by default.

The binary communication provides better performance and can be used by SAS Viya 3.5 or SAS 9.4. Open source clients such as Python SWAT require C language libraries to use the binary connection. Refer to the documentation for the open source client for information about the libraries.

If you enable either of these services, they are enabled as NodePorts by default. To use the services as LoadBalancers, you must specify LoadBalancer as the type. You can also restrict traffic by setting ranges of IP addresses for the load balancers to accept traffic on.

Note: The CAS operator supports setting the binary and HTTP services to either NodePort or LoadBalancer. Setting a combination of service types is not supported by the operator. In addition, the DC and EPCS services that are part of SAS/ACCESS and Data Connectors are also affected.

Configuration

- 1 Copy the `$deploy/sas-bases/examples/cas/configure/cas-enable-external-services.yaml` to your `$deploy/site-config` directory.
- 2 In the copied file, set the `publishBinaryService` key to true to enable binary communication for clients from outside the Kubernetes cluster:


```
- op: replace
  path: /spec/publishBinaryService
  value: true
```
- 3 If you want to enable the HTTP service, set the `publishHTTPService` key to true. This enables a service for REST access from outside the Kubernetes cluster. Be aware that REST access is enabled by default through a Kubernetes Ingress. If you have access through the Ingress, then enabling this HTTP service is redundant.


```
- op: replace
  path: /spec/publishHTTPService
  value: true
```
- 4 The services are configured as NodePort by default. For deployments in Microsoft Azure or Amazon Web Services (AWS), NodePort is not supported and you must configure the services as LoadBalancer services.

To configure them as LoadBalancer services, uncomment the `serviceTemplate`. Setting source ranges is optional. Delete the lines if you do not want them. Here is an example:

```
- op: add
  path: /spec/serviceTemplate
```



```

value:
  spec:
    type: LoadBalancer
    loadBalancerSourceRanges:
      - 192.168.0.0/16
      - 10.0.0.0/8

```

Note: SAS supports setting the `type` and `loadBalancerSourceRanges` keys in the service specification. Adding any other key such as `port` or `selector` can result in poor performance or prevent connectivity.

- 5 Set the `publishExtHostnameSuffix` key if you set the service to use `LoadBalancer`, your deployment is in Microsoft Azure or in AWS, and you meet either of these conditions:

- if you are using the DC or EPCS service.
- if the deployment is configured to use TLS.

When you set the key, the CAS Operator adds a subject alternative name (SAN) for each service to the certificate that is created by `sas-certframe`. The operator also adds a DNS label annotation to the service.

```

- op: add
  path: /spec/publishExtHostnameSuffix
  value: "-unique-name.subdomain-name"

```

For Microsoft Azure, replace *subdomain-name* with your Azure region name, such as `eastus2.cloudapp.azure.com`. The text in the value, up to the first period, is appended to the service name to create a unique DNS name. For example, the default value for the binary service is `sas-cas-server-default-bin`. If `-orion.eastus2.cloudapp.azure.com` is specified, then the operator creates the following annotation and publishes a DNS record for it.

```

apiVersion: v1
kind: Service
metadata:
  annotations:
    service.beta.kubernetes.io/azure-dns-label-name: sas-cas-server-default-bin-orion
...

```

For the example, the DNS record is `sas-cas-server-default-bin-orion.eastus2.cloudapp.azure.com`.

For AWS, replace *subdomain-name* with your subdomain of choice, such as `viya.acme.com`. For example the value you supply for `publishExtHostnameSuffix` could be `-pisces.viya.acme.com`. The `service.beta.kubernetes.io/azure-dns-label-name` annotation will be added to the deployment, but will be ignored by AWS. No DNS record will be generated. The administrator must create a DNS alias/CNAME record for each external service, including each node of the SAS Data Connect Accelerators, after deployment. See [“Configure External Access to Amazon Web Services CAS Services” on page 85](#) for details.

- 6 In the base `kustomization.yaml` file, add the path to your new `cas-enable-external-services.yaml` file to the `transformers` block:

```

transformers:
...
- site-config/{{ DIRECTORY-PATH }}/cas-enable-external-services.yaml

```

...

7 Save and close the kustomization.yaml file.

Note: If you configure direct access to the CAS server via HTTP or binary and are using full-stack TLS, the subject alternative names (SAN) in the certificate generated by cert-manager must include the host name or IP address being used to access that service. For the steps to include the host name or IP address, see [“Add the External Host Name or IP Address to the SAN in the Cert-Manager Generated Certificate”](#) in *Encryption in SAS Viya: Data in Motion*.

If you are making these changes after the initial deployment, the binary and HTTP services do not require that you restart CAS. For other services related to CAS, refer to the documentation to determine if a restart is required.

Use one of the next two sections to identify the connection information that programmers need to connect to CAS from outside the Kubernetes cluster.

Connection Information for Programmers: NodePort

You can use the following commands to identify the network port that maps to the service.

```
kubectl -n name-of-namespace get svc sas-cas-server-default-http
kubectl -n name-of-namespace get svc sas-cas-server-default-bin
```

For a NodePort, find the network port that programmers connect to.

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
sas-cas-server-default-bin	NodePort	10.0.5.236	<none>	5570:31066/TCP

Programmers need to know the host name of one of the Kubernetes nodes. You can use the following command to list the node names.

```
kubectl -n name-of-namespace get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
host02398.example.com	Ready	<none>	24d	v1.18.4
host02483.example.com	Ready	<none>	24d	v1.18.4
host02656.example.com	Ready	master	24d	v1.18.4
host02795.example.com	Ready	<none>	24d	v1.18.4
host02854.example.com	Ready	<none>	24d	v1.18.4

To connect from SAS Viya 3.5 or SAS 9.4 to the NodePort, run a CAS statement like the following example:

```
options CASHOST="host02398.example.com" CASPORT=31066;
cas casauto;
```

For the sas-cas-server-default-http service, a REST client connects to one of the Kubernetes nodes, such as host02398.example.com, and the port that is mapped to 8777.

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
sas-cas-server-default-http	NodePort	10.107.219.118	<none>	8777:31535/TCP

A REST client can connect to a resource such as `https://host02398.example.com:31535/cas-shared-default-http/`.

Connection Information for Programmers: LoadBalancer

You can use the following commands to identify the external IP address for the load balancer.

```
kubectl -n name-of-namespace get svc sas-cas-server-default-http
kubectl -n name-of-namespace get svc sas-cas-server-default-bin
```

The output includes the IP address of the load balancer and programmers connect to native port, 5570.

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
sas-cas-server-default-bin	LoadBalancer	10.0.44.57	52.247.0.1	5570:32215/TCP

To connect from SAS Viya 3.5 or SAS 9.4 to the LoadBalancer, run a CAS statement like the following example:

```
options CASHOST="sas-cas-server-default-bin-
orion.eastus2.cloudapp.azure.com";
options CASPORT=5570;

cas casauto;
```

Substitute your deployment-specific information for the sample unique value, orion, and the sample region, eastus2.

For the `sas-cas-server-default-http` service, a REST client connects to the load balancer on port 80 for HTTP or port 443 for HTTPS if TLS is configured. Only one of the two ports is operational, depending on whether TLS is configured.

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
sas-cas-server-default-http	LoadBalancer	10.0.61.68	52.247.0.168	8777:31979/ TCP,80:30707/TCP,443:30032/TCP

A REST client can connect to a resource such as `https://sas-cas-server-default-http-orion.eastus2.cloudapp.azure.com:443/cas-shared-default-http/`.

SAS Data Connect Accelerators

The SAS Data Connect Accelerators enable parallel data transfer between a distributed CAS server (MPP) and some data sources such as Teradata and Hadoop. For information about enabling connectivity for SAS/ACCESS and Data Connectors, see the README file at `$deploy/sas-bases/examples/data-access/README.md` (for Markdown) or `$deploy/sas-bases/docs/configuring_sasaccess_and_data_connectors_for_sas_viya_4.htm` (for HTML).

More Documentation

For more information about the services, see [“Kubernetes Services for CAS” in SAS Viya Operations: Servers and Services](#).

Enable Host Launch

By default, CAS cannot launch sessions under a user's host identity. All sessions run under the cas service account instead. CAS can be configured to allow for host identity launches by including a patch transformer in the kustomization.yaml file. To enable host launch for CAS, see the “Enable Host Launch in the CAS Server” section of the README file located at `$deploy/sas-bases/examples/cas/configure/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configuration_settings_for_cas.htm` (for HTML format).

Enable State Transfer for CAS Servers

Enabling state transfers preserves the sessions, tables, and state of a running CAS server for a new CAS server instance that is being started as part of a CAS server update. To enable the state transfer of CAS servers, see the README at `$deploy/sas-bases/overlays/cas-server/state-transfer/README.md` (for Markdown format) or `$deploy/sas-bases/docs/state_transfer_for_cas_server_for_sas_viya.htm` (for HTML format).

Note: You cannot enable state transfer and CAS auto-restart in the same SAS Viya deployment. If you want to enable state transfer for a deployment that already has CAS auto-restart enabled, you must first disable CAS auto-restart before enabling state transfer.

Enable CAS Auto-Restart After Updates

Note: If you are not using the SAS Viya Deployment Operator to manage your deployment, skip this section.

By default, CAS does not automatically restart during version updates performed by the SAS Viya Deployment Operator. To change the default to enable auto-restart, see the “CAS Auto-Restart During Version Updates” section of the README file located at `$deploy/sas-bases/overlays/cas-server/README.md` (for Markdown format) or `$deploy/sas-bases/docs/cas_server_for_sas_viya.htm` (for HTML format).

Note: You cannot enable CAS auto-restart and state transfer in the same SAS Viya deployment.

Create a Personal CAS Server

For development purposes in applications such as SAS Studio, you might need to allow data scientists the ability to work with a CAS server that is local to their SAS Compute session. This personal CAS server is just like a regular (shared) CAS server except it is simpler, relatively short-lived, and is only for one person.

To set up a personal CAS server, see the README file at `$deploy/sas-bases/overlays/sas-programming-environment/personal-cas-server/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configuring_sas_compute_server_to_use_a_personal_cas_server.htm` (for HTML format).

To set up a personal CAS server that uses a GPU, see the README file at `$deploy/sas-bases/overlays/sas-programming-environment/personal-cas-server-with-gpu/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configuring_sas_compute_server_to_use_a_personal_cas_server_with_gpu.htm` (for HTML format).

To configure a personal CAS server with or without a GPU, see the README file at `$deploy/sas-bases/examples/sas-programming-environment/personal-cas-server/README.md` (for Markdown format) or at `configuration_settings_for_the_personal_cas_server.htm` (for HTML format).

Configure SAS/CONNECT Settings

Support External Sign-on

To enable NodePort or LoadBalancer, see the README file at `$deploy/sas-bases/examples/sas-connect-spawner/README.md` (for Markdown language) or `$deploy/sas-bases/docs/configure_sasconnect_spawner_in_sas_viya.htm` (for HTML).

Note: In managed environments like Microsoft Azure, you cannot access the NodePort service from a client outside of the cluster.

Spawn SAS/CONNECT Servers Within the Spawner Pod

By default, you cannot spawn SAS/CONNECT servers within the spawner pod. However, this capability can be enabled by applying the security settings in the `enable-spawned-servers.yaml` example file. For the steps to enable spawning servers within the spawner pod, see the “Allow the Ability to Spawn Servers within the Spawner Pod” section of the “Configure SAS/CONNECT Spawner in SAS Viya” README file located at `$deploy/sas-bases/examples/sas-connect-spawner/README.md` (for Markdown format) and at `$deploy/sas-bases/docs/configure_sasconnect_spawner_in_sas_viya.htm` (for HTML format).

Connection Information for Programmers: NodePort

To sign on when a NodePort is specified:

- 1 Get the NodePort value that is mapped to the service port.

```
kubectl describe service/sas-connect-spawner-nodeport
```

or

```
kubectl get service/sas-connect-spawner-nodeport -o yaml
```

```
- name: service
  nodePort: 24133 // port that is exposed externally
  port: 17551
  protocol: TCP
```

- 2 Determine the host name of one of the Kubernetes nodes. If you are using no TLS or using TLS with self-signed certificates with the nodes in the DNS list or using a wildcard to match the nodes, you can use the following command to list the node names.

```
kubectl -n name-of-namespace get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
host02398.example.com	Ready	<none>	24d	v1.18.4
host02483.example.com	Ready	<none>	24d	v1.18.4
host02656.example.com	Ready	master	24d	v1.18.4
host02795.example.com	Ready	<none>	24d	v1.18.4
host02854.example.com	Ready	<none>	24d	v1.18.4

If you are using TLS with a cert-manager (such as `sas-viya-issuer`), `sas-certframe` adds the node name that the pod is running on to the certificate for the service. Use this command to find the log entry describing the addition of the node name to the certificate:

Note: Because of the length of the command and the margin of the page, this command appears as more than one line. The command should be entered as a single line.

```
kubectl -n name-of-namespace logs deployment/sas-connect-spawner sas-  
certframe | grep KUBE_NODE_NAME
```

In the output, the node name is listed and can be provided to programmers. Here is an example:

```
2020-09-03 23:11:45 - [INFO] - Adding KUBE_NODE_NAME host02398.example.com to  
SAS_CERTIFICATE_SAN_DNS
```

3 Sign on from an external client machine.

```
%let rem=node-name-from-step-2 nodeport-from-step-1;  
signon rem user='user-ID' password='password';
```

Using the examples from step 1 and 2, the command would look like this:

```
%let rem=host02398.example.com 24133;  
signon rem user='myuserid' password='mypassword';
```

Connection Information for Programmers: LoadBalancer

- 1 Determine the DNS name for the IP address that was provided by the load balancer. If you have not already registered a DNS name, you should do so now. The requirement to register a DNS name is described at [“Kubernetes Cluster Requirements” in System Requirements for SAS Viya](#).

Note: For information about DNS names while using Azure, see [Apply a DNS label to the service](#).

2 Sign on from an external client machine.

```
%let rem=DNS-name-from-step-1 17551;  
signon rem user='user-ID' password='password';
```

Configure External Storage Class for SAS Programming Run-Time Environment

All SAS Viya Servers

The Batch Server, Compute Server, and SAS/CONNECT Server are SAS Viya servers that use the SAS Programming Run-time Environment. They create a number of temporary files for run-time information in a location that is local to the `sas-programming-environment` pod. By default, these pods are backed by an `emptyDir` volume named `viya`, which is mounted automatically. However, using the default `emptyDir` volume is not recommended because SAS programming components can consume large amounts of storage quickly and cause nodes to shut down.

To configure different storage classes for the `viya` volume, see the README file at `$deploy/sas-bases/examples/sas-programming-environment/storage/README.md` (for Markdown format) or `$deploy/sas-bases/docs/sas_programming_environment_storage_tasks.htm` (for HTML format).

Batch Server Only

If you want the Batch Server to have storage that is different than the Compute Server and the SAS/CONNECT Server, such as using persistent storage rather than ephemeral storage, see the README file at `$deploy/sas-bases/examples/sas-batch-server/storage/README.md` (for Markdown format) or `$deploy/sas-bases/docs/sas_batch_server_storage_task_for_checkpoint_restart.htm` (for HTML format).

Configure GPUs

The SAS GPU Reservation Service aids SAS processes in resource sharing and utilization of the Graphic Processing Units (GPUs) that are available in a Kubernetes Pod. It is required in every SAS Cloud Analytic Services (CAS) Pod that is GPU-enabled. For information about implementing the SAS GPU Reservation Service, see the README located at `$deploy/sas-bases/examples/gpu/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/sas_gpu_reservation_service.htm` for HTML format.

Set Default SAS LOCALE and ENCODING in SAS Launcher Service

Setting the default locale and encoding for the SAS Launcher Service controls the default SAS LOCALE and ENCODING for SAS Compute Server, SAS/CONNECT, and SAS Batch Server, unless overridden by another specification. In order to set or modify these settings, see the “Locale and Encoding Defaults” section of the README file at `$deploy/sas-bases/examples/sas-launcher/configure/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configuration_settings_for_sas_launcher_service.htm` (for HTML format).

Change the Location of the NFS Server

SAS provides a transformer that allows you to change the location of the NFS server hosting the user's home directories. For information about using the transformer, see the “NFS Server Location” section of the README file located at `$deploy/sas-bases/examples/sas-launcher/configure/README.md` (for Markdown format) or `$deploy/sas-bases/docs/configuration_settings_for_sas_launcher_service.htm` (for HTML format).

Enable Access Methods Through LOCKDOWN System Option

SAS Viya uses the LOCKDOWN system option to limit access to files and features. By default, the following methods cannot be used to access files and specific SAS features for a SAS session that is executing in batch mode or server processing mode:

- EMAIL
- FTP
- HADOOP
- HTTP
- PYTHON
- PYTHON_EMBED
- SOCKET
- TCPIP
- URL

To enable any of these access methods, see the README file at `$deploy/sas-bases/examples/sas-programming-environment/lockdown/README.md` (for Markdown format) or `$deploy/sas-bases/docs/`

`lockdown_settings_for_the_sas_programming_environment.htm` (for HTML format). For more information about the LOCKDOWN system option, see [“LOCKDOWN System Option” in SAS Viya: Programming Run-Time Servers](#).

Configure SSSD

Enable SSSD

The configuration of SSSD is, by default, performed automatically for you. The automatic process uses the configuration service LDAP settings (if they exist) to construct an `sssd.conf` file. However, until that configuration is enabled, SSSD will not be available to your SAS Viya deployment. To enable that SSSD configuration:

- 1 Add a reference to the `sas-bases/overlays/cas-server/cas-sssd-sidecar.yaml` file to the transformers block in the base `kustomization.yaml` file. The new line must precede any lines for TLS transformers and the line for required transformers. Here is an example:


```
transformers:
- sas-bases/overlays/cas-server/cas-sssd-sidecar.yaml
...
- sas-bases/overlays/required/transformers.yaml
...
```
- 2 Follow the steps described in the “Disable Cloud Native Mode” in the “Configuration Settings for CAS” README file. The README is located at `$deploy/sas-bases/examples/cas/configure/README.md` (for Markdown format) and `$deploy/sas-bases/docs/configuration_settings_for_cas.htm` (for HTML format).
- 3 Because SSSD requires host authentication, follow the steps described at [“Enable Host Launch” on page 56](#).
- 4 Save and close the `kustomization.yaml` file.

Add a Custom Configuration for SSSD

If you would prefer to use a custom configuration for SSSD instead of the default, after completing the steps in [“Enable SSSD” on page 62](#), perform the following steps:

- 1 Copy the `$deploy/sas-bases/examples/cas/configure/cas-sssd-example.yaml` file to the location of your CAS server overlay, such as `$deploy/site-config/cas-server/cas-sssd-example.yaml`.
- 2 Add the location of the copied `cas-sssd-example.yaml` to the transformers block of the base `kustomization.yaml` file. The new line should go after the required transformers line. Here is an example based on the example used in step 1:

```
transformers:
```

```
...
- sas-bases/overlays/required/transformers.yaml
- site-config/cas-server/cas-sssd-example.yaml
...
```

- 3 Create your `sssd.conf` file and add your custom SSSD configuration to it. SAS recommends putting the `sssd.conf` file in the `$deploy/site-config` directory.
- 4 Add the following code to the `secretGenerator` block of the base `kustomization.yaml` file using the path to the `sssd.conf` file you created in step 3. Here is an example using `$deploy/site-config/cas-server/sssd.conf` as that path:

```
secretGenerator:
...
- name: sas-sssd-config
  files:
    - SSSD_CONF=site-config/cas-server/sssd.conf
  type: Opaque
...
```

- 5 Save and close the `kustomization.yaml` file.

Specify PersistentVolumeClaims to Use ReadWriteMany StorageClass

The manifest file that the base `kustomization.yaml` creates must have information about which PVCs in your deployment should take advantage of the `StorageClass` you created for your cloud provider.

- 1 In the `$deploy/site-config` directory, create a file named `storageclass.yaml`. Use the following content in that file.

```
kind: RWXStorageClass
metadata:
  name: wildcard
spec:
  storageClassName: {{ RWX-STORAGE-CLASS }}
```

Replace `{{ RWX-STORAGE-CLASS }}` with the name of your cluster's `StorageClass` that provides `ReadWriteMany` (RWX) access.

Note: If you are deploying on Microsoft Azure, replace `{{ RWX-STORAGE-CLASS }}` with the `sas-azurefile` class `StorageClass` you created at [“Additional PVC Requirements for Microsoft Azure”](#) in *System Requirements for SAS Viya*.

- 2 In the base `kustomization.yaml` file, add a `patches` block with the following content.

Note: The `annotationSelector` line in the following code is too long for the width of the page. A line break has been added to address the issue. If you copy this code for use, be sure to remove the line break.

```
patches:
```

```
- path: site-config/storageclass.yaml
  target:
    kind: PersistentVolumeClaim
    annotationSelector: sas.com/component-name in (sas-backup-job,sas-
data-quality-services,sas-commonfiles,sas-cas-operator,sas-pyconfig)
```

Note: If you are using the example kustomization.yaml file included at [“Initial kustomization.yaml File” on page 26](#), the patches block is already present.

Depending on the software that you are deploying, you might have to add more content to the annotationSelector line.

- If your order contains SAS Model Risk Management, add `sas-risk-cirrus-search,sas-risk-cirrus-builder` to the parenthetical list in the annotationSelector value.
- If your order includes SAS Risk Modeling, add `sas-risk-modeling-core` to the parenthetical list in the annotationSelector value.

Note: If you have changed the accessMode for CAS per the instructions at [“Change accessMode” on page 43](#), remove `sas-cas-operator` from the parenthetical list in the annotationSelector value.

Configure SAS/ACCESS

To configure and deploy your SAS/ACCESS products, see the “Configuring SAS/ACCESS and Data Connectors for Viya 4” README file at `$deploy/sas-bases/examples/data-access/README.md` (for Markdown) or `$deploy/sas-bases/docs/configuring_sasaccess_and_data_connectors_for_sas_viya_4.htm` (for HTML).

Create the SASDeployment Custom Resource

Note: If you are not using the SAS Deployment Operator, skip this section.

Install the Orchestration Tool

To create and maintain the SASDeployment custom resource, you must deploy the orchestration tool. If you have not already deployed the orchestration tool, do it now.

Follow the instructions in the “Prerequisites” section of the README file at `$deploy/sas-bases/examples/kubernetes-tools/README.md` (for Markdown format) or `$deploy/sas-bases/docs/using_kubernetes_tools_from_the_sas-orchestration_image.htm` (for HTML format).

Add an imagePullSecret for the SAS Viya Namespace

Depending on how you have planned your deployment, you might need to create an imagePullSecret for the namespace in which you are deploying SAS Viya. If the SAS Viya content has been mirrored and the mirror requires authentication, you must create an imagePullSecret. The secret must be named `sas-orchestration-secret` and can be added with a `kubectl` command:

```
kubectl -n name-of-namespace \
  create secret generic sas-orchestration-secret \
  --type=kubernetes.io/dockerconfigjson \
  --from-file=.dockerconfigjson=file-with-secret-content
```

For example, if you are deploying SAS Viya into a namespace called `viya`, and the secret content is in a file named `site-config/image-pull-secret.json`, the command would look like this:

```
kubectl -n viya \
  create secret generic sas-orchestration-secret \
  --type=kubernetes.io/dockerconfigjson \
  --from-file=.dockerconfigjson=site-config/image-pull-secret.json
```

Note: For more information about the command to create imagePullSecrets, see [Pull an Image from a Private Registry](#).

Run the create sas-deployment-cr Command

Note: The container to create the custom resource runs under the `sas` ID and group. Ensure that the `$(pwd)` directory, specified in the `create` command, has permissions that can accommodate the `sas` ID and group.

As an administrator with local cluster permissions, run the following command to create the SASDeployment custom resource and to name it `$deploy-sasdeployment.yaml`. Make sure that the command is run from the parent directory of the `$license` and `$deploy` directories. Here is the command format:

```
docker run --rm \
  -v $(pwd):mount-for-working-directory-inside-container \
  sas-orchestration \
  create sas-deployment-cr \
  --deployment-data certificates-information \
  --license license-information \
```

```
--user-content location-of-deployment-files \
--cadence-name cadence-type \
--cadence-version cadence-version-number \
[--cadence-release cadence-release-number \]
[--image-registry mirror-registry-location \]
[--repository-warehouse repository-warehouse-location \]
> $deploy-sasdeployment.yaml
```

Here is a description of the values to be substituted for the variables in the command:

Note: For information about all the flags available for the `create sas-deployment-cr` command, use the help flag:

```
docker run --rm \
sas-orchestration \
create sas-deployment-cr \
--help
```

mount-for-working-directory-inside-container

The path at which the current working directory should be mounted inside the container.

certificates-information

The location of the `*-certs.zip` file. It can be a directory path, which includes the mount for the working directory, or a go-getter URL.

license-information

The location of the license, which can be a directory path, including the mount for the working directory, or a go-getter URL.

location-of-deployment-files

The location of the `$deploy` directory. This can be a directory path, including the mount for the working directory, or a go-getter URL.

cadence-type

Use `stable` for software in the Stable cadence, or use `lts` for the Long-Term Support cadence.

cadence-version-number

The cadence version number of the software to be deployed (for example, 2020.1.4).

[cadence-release-number] (optional)

The latest cadence release or a specific cadence release of the cadence version number of the software to be deployed. See the important note that follows.

[mirror-registry-location] (optional)

The URL for the docker image registry (for example, `registry.example.com`). This flag is needed if you are deploying with a mirror registry.

[repository-warehouse-location] (optional)

The URL for the warehouse describing what should be deployed. This flag is needed if you are managing a dark environment.

\$deploy

Precede the name of the `sasdeployment.yaml` file with the name of the directory that the software is being deployed from. For example, if you use `viyal` as `$deploy`, the file should be named `viyal-sasdeployment.yaml`.

Note: The files being pulled into the custom resource must be text files. If you must use a binary file, it should be added to the custom resource by using a go-getter URL. For information about go-getter URLs, see <https://github.com/hashicorp/go-getter>.

IMPORTANT If you specify the cadence release, its specification affects how the operator reconciles the custom resource. Consider the following when deciding how to use the `--cadence-release` option.

- If you are using a mirror registry and used a cadence release to populate it, the value in the CR must match that cadence release.
- Set the value to "" in order to force the operator to use the latest release for the requested *cadence-version-number*. A consequence of using this value is that processes that seem unrelated to software updates, such as renewing your license or making configuration changes, might also spawn updates to the running software.
- Set the value to a specific cadence release value to use that cadence release in the custom resource. If the specified cadence release does not exist, the user is presented with an error.
- If no cadence release is specified, as in the examples that follow, the operator uses the latest cadence release when the custom resource is initially introduced to a namespace. The operator automatically assigns the chosen release value in the cluster representation of this custom resource. If the user later applies an updated custom resource, without a cadence release specified, Kubernetes preserves the previously assigned value of this field and therefore the versions of any software already deployed into that namespace. To change the version of software in an existing namespace, assign a cadence release as described above.

Here is an example of the command with the following values.

- The directory should be mounted in `/cwd/` in the container.
- The `*-certs.zip` file is located at `/cwd/license/SASViyaV4_69SWC4_certs.zip`.
- The license file from SAS is located at `/cwd/license/SASViyaV4_69SWC4_lts_2021_license_2020-09-08T105930.jwt`.
- The `$deploy` directory is `/cwd/viya1`.
- The software being deployed is Long-Term Support 2021.1.

```
docker run --rm \
-v $(pwd):/cwd/ \
sas-orchestration \
create sas-deployment-cr \
--deployment-data /cwd/license/SASViyaV4_69SWC4_certs.zip \
--license /cwd/license/SASViyaV4_69SWC4_lts_2021_license_2020-09-08T105930.jwt \
--user-content /cwd/viya1 \
--cadence-name lts \
--cadence-version 2021.1 \
> viya1-sasdeployment.yaml
```

Here is an excerpt of the generated custom resource:

...

```

---
apiVersion: orchestration.sas.com/v1alpha1
kind: SASDeployment
metadata:
  name: sas-viya
spec:
  cadenceName: lts
  cadenceVersion: "2021.1"
  license:
    secretKeyRef:
      name: sas-viya
      key: license
  clientCertificate:
    secretKeyRef:
      name: sas-viya
      key: cert
  caCertificate:
    secretKeyRef:
      name: sas-viya
      key: cacert
  userContent:
    files:
      kustomization.yaml: |
        resources:
          - sas-bases/base
          - sas-bases/overlays/cert-manager-issuer
...

```

Notice that the values that were entered in the command are included in the custom resource. The custom resource also includes a transcription of the contents of the base kustomization.yaml file.

Here is an example of the command that uses references (in the form of go-getter URLs) to the locations for the values:

- The directory should be mounted in `/cwd/deplo`y in the container.
- The `*-certs.zip` file is located at `https://example.com/SASViyaV4_69SWC4_certs.zip`.
- The license file from SAS is located at `https://example.com/SASViyaV4_69SWC4_lts_2021_license_2020-09-08T105930.jwt`.
- The `$deploy` directory is `git::https://user:token@git.example.com/repository.git//viya1`.
- The software that is being deployed is Long-Term Support 2021.1.

Note: When fetching from a Git repository, in order for the content to be cloned locally by the operator before being used, you must use the annotation `environment.orchestration.sas.com/readOnlyRootFilesystem: "false"`.

```

docker run --rm \
  -v $(pwd):/cwd/deplo \
  sas-orchestration \
  create sas-deployment-cr \
  --deployment-data https://example.com/SASViyaV4_69SWC4_certs.zip \
  --license https://example.com/SASViyaV4_69SWC4_lts_2021_license_2020-09-08T105930.jwt \
  --user-content git::https://user:token@git.example.com/repository.git//viya1 \

```



```
--cadence-name lts \
--cadence-version 2020.1 \
> viyal-sasdeployment.yaml
```

The generated custom resource would include this content:

```
...
---
apiVersion: orchestration.sas.com/v1alpha1
kind: SASDeployment
metadata:
  annotations:
    environment.orchestration.sas.com/readOnlyRootFilesystem: "false"
  creationTimestamp: null
  name: sas-viya
spec:
  caCertificate:
    url: https://example.com/SAS_CA_Certificate.pem
  cadenceName: lts
  cadenceVersion: "2020.1"
  clientCertificate:
    url: https://example.com/entitlement_certificate.pem
  license:
    url: https://example.com/SASViyaV4_69SWC4_lts_2021_license_2020-09-08T105930.jwt
  repositoryWarehouse: {}
  userContent:
    url: git::https://user:token@git.example.com/repository.git//viyal
status:
...
```

Notice that the custom resource contains the information that is included in the command.

Note: For more information about the fields in the SASDeployment custom resource, see [“Fields in the SASDeployment Custom Resource” on page 120](#).

Deploy the Software

Deployment Using the SAS Viya Deployment Operator

Command and Output

Because the operator is actually running as a result of the last command that you performed in [“Apply the SAS Viya Deployment Operator Resources to the Cluster”](#)

on page 13, the operator responds to any changes to the SASDeployment custom resource by applying those changes. Therefore, to perform the initial deployment, run the following command as an administrator with local cluster permissions to apply the SASDeployment custom resource:

```
kubectl -n name-of-namespace apply -f $deploy-sasdeployment.yaml
```

Note: SAS recommends that you save a copy of the SASDeployment custom resource locally or to Git as a backup.

To determine the status of the deployment, run the following command:

```
kubectl -n name-of-namespace get sasdeployment
```

Here is an example of the output:

NAME	STATE	CADENCENAME	CADENCEVERSION	CADENCERELEASE	AGE
viya1 130m	SUCCEEDED	stable	2020.1.3	20210304.1614817334881	

The `STATE` field cycles through several values. The field value starts with `PENDING`, then `RECONCILING`, and finishes in either `SUCCEEDED` or `FAILED`. For more information about communications from the SAS Viya Deployment Operator, see [“Communications from the Operator” on page 125](#).

Initial Troubleshooting

When the SAS Viya Deployment Operator is not working as expected, three different sources can be used to diagnose problems. If you need to contact SAS Technical Support for help, be sure to share the output from all three of these sources.

Log from the SAS Viya Deployment Operator Pod

The log from the SAS Viya Deployment Operator pod can be useful in diagnosing problems that might be preventing the SAS Viya Deployment Operator from deploying SAS Viya. By default, that pod is named `sas-deployment-operator-hash`. The Kustomize tool appends the hash value during the deployment of the SAS Viya Deployment Operator. An example pod name is `sas-deployment-operator-57f567f7bc-drg5z`.

Use the following command to generate log output:

```
kubectl \
  logs \
    -n name-of-deployment-operator-namespace \
    deployment-operator-pod-name
```

SASDeployment Custom Resource

The `.status` field of a SASDeployment custom resource contains information about the last attempt to deploy SAS Viya. For complete details about this field, see [“Communications from the Operator” on page 125](#). Specifically, the `.status.messages` field contains all the messages from the last Reconcile Job

that was started by the SAS Viya Deployment Operator. These messages relate to fetching URLs, running Kustomize, and running kubectl.

Use the following command to generate output for the entire SASDeployment custom resource:

```

kubectl \
  get sasdeployments \
  -n name-of-SAS-Viya-namespace \
  -o yaml

```

Log from the Reconcile Job

The log from the SAS Viya Deployment Operator Reconcile Job can be useful in diagnosing problems with deploying a particular SASDeployment custom resource. By default, that Job is named `sas-deployment-operator-reconcile-hash`. A unique Job is associated with each deployment attempt. All these Jobs are located in the same namespace as the SASDeployment custom resource that they are deploying, providing a historical record of those attempts. The Jobs are removed automatically after the associated SASDeployment custom resource is removed from Kubernetes.

Depending on cluster settings, the pod that is run by the Job might not be available after the process exits. However, if the pod remains, use the following command to generate output for its Job log:

```

kubectl \
  logs \
  -n name-of-SAS-Viya-namespace \
  reconcile-Job-pod-name

```

Remediation

If an issue prevents the successful deployment of your software and one of the sources described above indicates the issue is associated with content in `$deploy/site-config` or the base `kustomization.yaml` file, take the following steps to address the issue before contacting SAS Technical Support:

- 1 Make corrections for the error. Debugging can include reviewing example files for formatting, file names, or path specifications. The base `kustomization.yaml` file can also be reviewed to ensure it was revised as necessary. To help with debugging, refer to the appropriate documentation, including README files.
- 2 Rebuild the SASDeployment custom resource using the instructions at [“Run the create sas-deployment-cr Command” on page 65](#).
- 3 Apply the custom resource using the instructions at [“Command and Output” on page 69](#).

Manual Deployment

Note: If you have deployed the SAS Viya Deployment Operator, these commands are not necessary since the operator deploys your software for you. For more

information, see `$deploy/sas-bases/examples/deployment-operator/deploy/README.md` (for Markdown) or `$deploy/sas-bases/docs/sas_viya_deployment_operator.htm` (for HTML).

- 1 On the kubectl machine, create the Kubernetes manifest:

```
kustomize build -o site.yaml
```

The following message might be displayed:

```
well-defined vars that were never replaced: SAS_COMPONENT_RELPATH_sas-orchestration,SAS_COMPONENT_TAG_sas-orchestration
```

If this message is displayed, it can safely be ignored.

- 2 Apply cluster-api resources to the cluster. As an administrator with cluster permissions, run

```
kubectl apply --kubeconfig=kubeconfig-file --selector="sas.com/admin=cluster-api" -f site.yaml
```

```
kubectl wait --kubeconfig=kubeconfig-file --for condition=established --timeout=60s -l "sas.com/admin=cluster-api" crd
```

The kubectl apply command might cause a the following message to be displayed:

```
error: no objects passed to apply
```

If this message is displayed, it can safely be ignored.

- 3 As an administrator with cluster permissions, run

```
kubectl apply --kubeconfig=kubeconfig-file --selector="sas.com/admin=cluster-wide" -f site.yaml
```

The kubectl apply command might cause the following message to be displayed:

```
unable to recognize "site.yaml": no matches for kind "foo" in version "bar"
```

If this message is displayed, it can safely be ignored.

- 4 As an administrator with local cluster permissions, run

```
kubectl apply --kubeconfig=kubeconfig-file --selector="sas.com/admin=cluster-local" -f site.yaml --prune
```

- 5 As an administrator with namespace permissions, run

```
kubectl apply --kubeconfig=namespace-kubeconfig-file --selector="sas.com/admin=namespace" -f site.yaml --prune
```

The kubectl apply command might cause the following message to be displayed:

```
error: error pruning nonNamespaced object
```

If this message is displayed, it can safely be ignored.

- 6 If you are performing an update, as an administrator with namespace permissions, run the following command to prune additional resources not in the default set.

```
kubectl apply --kubeconfig=namespace-kubeconfig-file --selector="sas.com/admin=namespace" -f site.yaml --prune --prune-
whitelist=autoscaling/v2beta2/HorizontalPodAutoscaler
```

- 7 Wait for Kubernetes to create and start the pods. To determine whether the pods have started:

```
kubectl --kubeconfig=namespace-kubeconfig-file -n name-of-namespace get pods
```

The output of this command looks like this:

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
dl0006	annotations-66dc4479fd-qfqqr	1/1	Running	0	5s
dl0006	appregistry-bbbdfb78c-tcllv	1/1	Running	0	5s
dl0006	audit-7c4ff4b8b8-zxg8k	1/1	Running	0	5s
dl0006	authorization-79d4f594b9-t9sbx	1/1	Running	0	5s
dl0006	cachelocator-668fcd544-hcxbs	1/1	Running	0	5s
dl0006	cacheserver-7dc898d4bf-8dfgx	1/1	Running	0	5s
dl0006	casaccessmanagement-64b5769d8f-mlmjf	1/1	Running	0	5s
dl0006	casadministration-747746f94c-j2dm2	1/1	Running	0	5s

During startup some pods restart a number of times until other pods are ready. The value in the **Status** column is Running or Completed when the pods have either fully started or completed their expected function.

Readiness Service

The readiness service checks the status of the SAS Viya deployment to determine whether it is ready for use. The service performs all of its checks every 30 seconds. After the software is deployed, the service should be consulted to determine whether the deployment is ready for use. The readiness service is also a useful tool for the administration of SAS Viya deployment throughout its life.

To see the results of the latest readiness check:

```
kubectl wait -n name-of-namespace \
  --for=condition=ready pod \
  --selector="app.kubernetes.io/name=sas-readiness" \
  --timeout=1800s
```

If the deployment is ready, the command has the return code value 0, and the following output is displayed:

```
pod/sas-readiness-hash condition met
```

If the deployment is not ready, the command has the return code value 1, and the following output is displayed:

```
pod/sas-readiness-hash condition met
error: timed out waiting for the condition on pod/sas-readiness-hash
```

If the deployment is not ready, use this command to check the logs of the readiness service to determine which service is failing:

Note: The command should be entered as a single line.

```
kubectl logs -n name-of-namespace --selector="app.kubernetes.io/name=sas-readiness"
```

Sign In as the sasboot User

Your SAS environment is deployed with an initial administrator account that is named `sasboot`. The password for this account has expired by default, so you must reset the password before you can sign in.

To reset the password:

- 1 Get the name of the pod that contains SASLogon and search for the characters, `sasboot`:

```
kubectl logs -n name-of-namespace -c sas-logon-app $(kubectl -n name-of-namespace get pods | grep sas-logon | cut -f1 -d' ' ) | grep sasboot
```

Note: This command will not return a value if you set the `sas.logon.initial.password` property in a `sitedefault.yaml` file.

- 2 Sign in from a URL with this format:

```
protocol://name-of-ingress-host:port/SASLogon/reset_password?code=password
```

`protocol` can be `http` or `https`, depending on how you have secured your deployment.

- 3 Follow the instructions on the displayed web page to reset the password.

If the URL has expired, restart the `sas-logon` pod:

```
kubectl -n name-of-namespace delete pod sas-logon-pod-name
```

Then go to the log and obtain the new URL. The URL expires 24 hours after the SAS Logon service restarts. For security purposes, the URL that is specified in a browser or in a text editor also expires, even if the password is not reset.

After you reset the password, SAS Environment Manager automatically opens in your browser.

- 4 Click **Yes** for all of the assumable groups so that you have the permissions to perform subsequent tasks.
- 5 Share the following URL with any other users of your SAS Viya software so that they can access the deployment:

```
protocol://name-of-ingress-host:port/SASDrive
```

`protocol` can be `http` or `https`, depending on how you have secured your deployment.

After the password has been reset, you should consider disabling the password reset feature. For the steps to disable that feature, see [“Disable the Password Reset Feature” in SAS Viya: Identity Management](#).

Note: If you have enabled multi-tenancy in your deployment, using the sasboot user to reset the password opens SAS Environment Manager on the provider tenant.

Promotion and Migration

If you plan to move pre-existing content from SAS 9 or SAS Viya 3, see [Full-System Migration and Content Migration](#).

Post-Installation Tasks

Configure Identities	78
Configure an LDAP Identity Provider	78
Configure a SCIM Identity Provider	78
Configure Multi-tenancy	79
Configure the Connection to the Mail Service	79
Configure Files Service	80
Configure Monitoring and Logging	80
Configure Guest Access	81
Obtain and Run Hadoop Tracer Script	81
Overview	81
Transfer the Hadoop Tracer Script to the Hive Node	82
Tasks Before Running the Hadoop Tracer Script	82
Runtime Options	83
Run the Hadoop Tracer Script	83
Determine If the Run Was Successful	84
How to Address Failures	84
Copy the JAR and Configuration Files to the SAS Client System	85
Configure Cloud Analytic Services (CAS)	85
Configure External Access to Amazon Web Services CAS Services	85
Configure Model Access	87
Configure Access to Models	87
Configure Access to Analytic Store Model Files	87
Configure SAS Data Quality	88
Configure the Quality Knowledge Base	88
Configure SAS Data Science Programming	88
Additional Software Associated with SAS Data Science Programming	88
Configure SAS Event Stream Processing	89
Configure SAS for Microsoft 365	89
Configure SAS Intelligent Decisioning	89
Configure SAS Model Manager	90

Configure SAS Risk Modeling	90
Configure SAS Visual Analytics	90
Additional Software Associated with SAS Visual Analytics	90
Configure SAS Visual Data Science	91
Additional Software Associated with SAS Visual Data Science	91
Configure SAS Visual Data Science Decisioning	91
Additional Software Associated with SAS Visual Data Science Decisioning	91
Configure SAS Visual Machine Learning	92
Additional Software Associated with SAS Visual Machine Learning	92
Configure SAS Visual Statistics	93
Additional Software Associated with SAS Visual Statistics	93
Configure SAS/CONNECT Spawner	93

Configure Identities

Configure an LDAP Identity Provider

After completing the installation of SAS Viya, you must configure an identity provider before your users can access SAS Environment Manager and SAS Visual Analytics. LDAP is the default identity provider. Make sure your LDAP server meets the requirements that are described in [LDAP Requirements](#).

For a full set of instructions for configuring an LDAP identity provider, see “[How to Configure LDAP](#)” in *SAS Viya: Identity Management*.

Configure a SCIM Identity Provider

After completing the installation of SAS Viya, you must configure the connection to your identity provider before your users can access SAS Environment Manager and other SAS Viya products. To use SCIM as your source for user and group identities, you must grant Microsoft Azure access to the SAS Viya SCIM endpoints.

Microsoft Azure

To complete SCIM identity provider configuration, you must configure the SCIM connection to SAS Viya in Microsoft Azure Active Directory. You also need to disable LDAP, which is configured by default when the deployment completes. For a full set of instructions, see “[How to Configure SCIM](#)” in *SAS Viya: Identity Management*.

Amazon Web Services

You can use [Amazon Web Services](#) to access the SAS Viya SCIM endpoints using SCIM as your source for user and group identities. SAS Viya should support authentication and identity management with Amazon Web Services Identity and Access Management, with only the initial configuration linking SAS Viya to AWS IAM. Authentication should be delegated to AWS and identity information should be synchronized into SAS Viya with the SCIM identity provider configuration.

Users can change the provisioning from manual to SCIM by following the instructions to enable automatic provisioning at [Enabling Single Sign-On Between OneLogin and AWS](#).

Google Cloud Platform


You can utilize [Google Cloud Platform](#) to access the SAS Viya SCIM endpoints to use SCIM as your source for user and group identities. To set up Google Cloud for SCIM, follow the instructions on the web page.

Configure Multi-tenancy




After deployment, a Kubernetes administrator with elevated permissions can onboard and offboard tenants, monitor services, monitor logs, stop and start tenants, and run multi-tenant backups and restores. SAS recommends onboarding at least one tenant into a multi-tenant environment immediately after deployment. For information about performing each of these tasks, see [SAS Viya: Multi-tenancy](#).

Configure the Connection to the Mail Service

After installing a new SAS Viya deployment, you must configure the connection to your mail service. Complete these steps while you are signed in as one of the SAS Administrators.

- 1 Select the  from the side menu to open the Configuration page.
- 2 On the Configuration page, select **All Services** from the list, and then select **Mail service** from the list of services.

Note: If **Mail service** is not listed, then there is no configuration to be performed. You should skip the rest of this topic.

- 3 In the **sas.mail** section, click . In the Edit Configuration window, follow these steps:
 - a Specify a value for the following required fields: **host** and **port**. For the remaining fields, review the default values and make changes, as necessary. The default values are appropriate for most sites.
 - b Click **Save**.
- 4 (Optional) To enable the health check for the mail service, perform the following steps.
 - a Select the  from the side menu to open the Configuration page.
 - b On the Configuration page, select **All Services** from the list, and then select **Mail service** from the list of services.
 - c In the **management.health.mail** section, click .
 - d Turn the **enabled** toggle to on.
 - e Click **Save**.

When this toggle is set, health checks will be enabled after the mail service is restarted. If the mail host is not configured or is configured incorrectly, or if it cannot connect to the SMTP mail server, the mail service will indicate it is in a failed state.

Configure Files Service

After deploying SAS Viya,, the Files service must be configured to provide in-line responses to file contents. For the steps to perform this configuration, see [“Avoid Cross-Site Scripting Vulnerabilities” in Encryption in SAS Viya: Data in Motion](#).

Configure Monitoring and Logging

Monitoring performance metrics and the logs generated by your deployment helps your SAS administrators ensure that SAS Viya is running efficiently and enable them to quickly detect any problems. You can use your preferred monitoring technology to monitor your SAS Viya deployment. If you do not have a preferred technology, SAS provides solutions for monitoring and logging that are based on widely-used open source monitoring technologies including Prometheus, Grafana, and Elasticsearch.

You can deploy these solutions from the [SAS Viya Monitoring for Kubernetes GitHub site](#).

The monitoring deployment includes these components:

- Prometheus
- Prometheus Operator
- Alert Manager
- Grafana
- Prometheus Pushgateway
- Grafana dashboards

The logging deployment includes these components:

- Fluent Bit
- Elasticsearch
- Kibana

Configure Guest Access

Guest access is an optional feature that provides anonymous Read-Only access to a subset of resources and functionality in participating applications. For information about configuring your SAS Viya deployment for guest access, see [“Authentication: Guest Access” in SAS Viya: Authentication](#).

Obtain and Run Hadoop Tracer Script

Note: The Hadoop tracer script is included with SAS/ACCESS Interface to Hadoop and SAS In-Database Technologies for Hadoop. The Hadoop tracer script might also be required by additional components that access Hadoop. If you do not need to run the Hadoop tracer script, then skip this section.

Overview

The Hadoop tracer script is a Python script that traces the system calls of various Hadoop client tools and uses the trace data to identify required client JAR files and configuration files. The script determines which specific Hadoop distribution supplied the JARs and configuration files that SAS products need for connectivity between Hadoop client machines and the Hadoop server environment. Therefore, you must run this script on the Hive node of your Hadoop cluster in order to prepare your Hadoop environment for a SAS Viya deployment.

The script is organized into two parts. The first part is a list of the Hadoop services that the script is tracing to a new `driver.json` file. The second part is the remainder of the code. This division allows the list of Hadoop services and additional JAR files to be modified manually without having to make Python code changes.

If the `driver.json` file is moved or removed from its default location, the Hadoop tracer script fails with an error message indicating that the file is missing.

Transfer the Hadoop Tracer Script to the Hive Node

The Hadoop tracer script runs on the Hive node of your Hadoop cluster.

- 1 On your Hadoop cluster, create a temporary directory to hold a ZIP file that you download later. For example, `/tmp/sas/hadooptracer`.

- 2 Copy and paste the following URL into a browser. Download the `hadooptracer.zip` file from that FTP site to the directory that you created in step 1.

`ftp.sas.com/techsup/download/blind/access/hadooptracer.zip`

- 3 Using a method of your choice (such as PSFTP, SFTP, SCP, or FTP), transfer the ZIP file to the Hive node on your Hadoop cluster.

- 4 Unzip the file. The `hadooptracer_py` and `driver.json` files are included in the ZIP file.

- 5 Change permissions on the `hadooptracer_py` file to include the Execute permission:

```
chmod 755 ./hadooptracer_py
```

Tasks Before Running the Hadoop Tracer Script

- Ensure that the user running the script has authorization to issue HDFS and Hive commands.
- If Hadoop is secured with Kerberos, obtain a Kerberos ticket for the user before running the script.
- Ensure that Python and the `strace` Linux library have been installed on the Hadoop cluster. Install them from the package repositories for your Linux distribution if necessary. Python 2.6 or later is required.
- If you want to pull one or more JAR files that are not included in the output of the Hadoop tracer script, modify the `driver.json` file by adding the JAR files to the `ExtraJarFiles` property at the bottom of the file. Here is an example of adding a new JAR file named `xyz-service.jar` to the `ExtraJarFiles` property:

```
"ExtraJarFiles": ["jline-*", "jruby-complete-*", "hive-warehouse-connector-assembly-
*", "xyz-service-*"]
```

Runtime Options

For a list of the options available when running the Hadoop tracer script:

```
python ~/hadooptracer_py --help
```

Review these options before you run the script.

Run the Hadoop Tracer Script

Use this command to run the Hadoop tracer script:

```
python ./hadooptracer_py --filterby=latest --postprocess --jsonfile ./driver.json
```

If you want to pull the necessary JAR files without filtering, use `filterby=none`, or omit the `filterby=` option.

TIP To collect only the Hadoop configuration files and exclude JAR files:

```
python ./hadooptracer_py --svckey=Hadoop
```

TIP The `postprocess` option ensures that the `${hdp.version}` tokens are replaced. SAS strongly recommends that you run the tracer script with this option. This option is ignored for Cloudera clusters.

The Hadoop tracer script performs the following tasks:

- collects the necessary Hadoop JAR and configuration files from nodes in the cluster and copies them in the `/tmp/jars` directory and the `/tmp/sitexmls` directory, respectively.
- creates a `hadooptracer.json` file in the `/tmp` directory. If you need a custom path for the JSON output file, use this command instead:

```
python ./hadooptracer_py -f /your-path/hadooptracer.json
```

- creates a log at `/tmp/hadooptracer.log`. If you need a custom path for the log file, add this option:

```
--logfile your-path/log-file-name
```

- prints more debug entries to the log file when the `--debug` option is added.

IMPORTANT The Hadoop JAR and configuration files on the SAS client machine must be kept in sync with the Hadoop configuration. After a Hadoop cluster update, re-run the Hadoop tracer script to collect any new files, and then copy those files to the SAS client machine, replacing any previous versions of the files.

Determine If the Run Was Successful

- 1 Ensure that the required Hadoop JAR files are collected from the Hadoop cluster and placed in the `./jars` directory.

```
ls -l ./jars
```

- 2 Ensure that the required Hadoop configuration files are collected from the Hadoop cluster and placed in `./configs` directory.

```
ls -l ./configs
```

- 3 Review the `hadooptracer.log` file that is located in the default location, `/tmp`, or the custom location that you specified.

How to Address Failures

Most errors with the Hadoop tracer script stem from improper usage or an incorrect cluster configuration. If there are a problems with the Hadoop cluster, they will typically show up in the stdout of the Hadoop tracer script in the form of Java traceback information.

Another common problem occurs when users try to run the Hadoop tracer script on a cluster node that doesn't have `Hadoop/Hive/HDFS/Yarn/Pig/etc` in an available `PATH`. For example,

```
2020-04-07 12:16:51,036 hadooptracer [ERROR] pig is not found in the $PATH
```

Inspect `hadooptracer.log`, located in `/tmp` by default, and use the rest of this troubleshooting section to resolve common issues. Some error messages in the console output for `hadooptracer_py` are normal and do not necessarily indicate a problem with the JAR and configuration file collection process. However, if the files are not collected as expected or if you experience problems connecting to Hadoop with the collected files, contact SAS Technical Support and include the `hadooptracer.log` and the `hadooptracer.json` files.

Copy the JAR and Configuration Files to the SAS Client System

SAS 9.4 and SAS Viya 3.5

- 1 On the SAS client machine, create two directories to hold the JAR and configuration files. For example, the `/opt/sas/hadoopfiles/lib` and `/opt/sas/hadoopfiles/conf` directories.
- 2 Using a method of your choice (such as PSFTP, SFTP, SCP, or FTP), copy the files in the `tmp/jars` and `tmp/conf`s directories on the Hadoop server to the directories on the SAS client machine that you created.

Note: If you connect to the Hadoop server with an HTTP REST API, you do not need the Hadoop JAR files on the SAS client machine.

SAS Viya in a Kubernetes Environment

For more information about configuring SAS/ACCESS Interface to Hadoop, see the README file at `$deploy/sas-bases/examples/data-access/README.md` (for Markdown) or at `$deploy/sas-bases/docs/configuring_sasaccess_and_data_connectors_for_sas_viya_4.htm` (for HTML).

Configure Cloud Analytic Services (CAS)

Configure External Access to Amazon Web Services CAS Services

To gain secure external access to CAS services running in an AWS cloud, additional steps are necessary after the load balancer setup described in [“Configure External Access to CAS” on page 51](#) has been completed and the cluster has been deployed.

All load balancer hosts must be DNS-aliased to the service names generated by setting up the load balancer. You may select an extant DNS domain/subdomain or create a new one specifically to support access to AWS CAS clusters. You may use

your own DNS service or transfer DNS management of the domain to the AWS Route 53 service.

Note: AWS Route 53 offers some advantages for large scale or more complex deployment cases due to the additional routing and monitoring capabilities that it offers. See [What is Amazon Route 53?](#) for details.

Regardless of the DNS service used, you must add CNAME records to the routing table to direct references to the CAS-generated names to the load balancer FQDN. CAS-generated external service names are predetermined based upon the suffix supplied in the earlier setup. Here are examples of those names when using Data Connect Accelerators and Spark EPCS access in addition to the general ports:

```
sas-cas-server-default-bin-pisces.viya.acme.com
sas-cas-server-default-epcs-pisces.viya.acme.com
sas-cas-server-default-controller-dc-pisces.viya.acme.com
sas-cas-server-default-backup-dc-pisces.viya.acme.com
sas-cas-server-default-worker-0-dc-pisces.viya.acme.com
sas-cas-server-default-worker-1-dc-pisces.viya.acme.com
```

To find the load balancer names:

```
kubectl -n name-of-namespace get svc | grep LoadBalancer
```

Here is a partial sample of typical output, revealing the load balancer DNS names:

```
sas-cas-server-default-backup-dc LoadBalancer 10.100.178.234
a7cc66edbcald402c8c72ea7da3543d7-1688256446.us-east-1.elb.amazonaws.com
sas-cas-server-default-bin LoadBalancer 10.100.169.27
a5a90408edfc246bba051f830d6e58fe-2016925591.us-east-1.elb.amazonaws.com
sas-cas-server-default-controller-dc LoadBalancer 10.100.30.89
a14ca0a2c8009415891c05053ee36d83-222471472.us-east-1.elb.amazonaws.com
sas-cas-server-default-epcs LoadBalancer 10.100.105.198
a8207fb65f9464e7680948e8e8d9e722-58303842.us-east-1.elb.amazonaws.com
sas-cas-server-default-http LoadBalancer 10.100.65.146
aa21afd024de44d64bffe557add1e47a-1604903159.us-east-1.elb.amazonaws.com
sas-cas-server-default-worker-0-dc LoadBalancer 10.100.4.214
a307341cce8694e639c6fdee9096cb5c-632571573.us-east-1.elb.amazonaws.com
sas-cas-server-default-worker-1-dc LoadBalancer 10.100.224.50
a04b03687b3704b79820fb72869eb5c5-1997141149.us-east-1.elb.amazonaws.com
```

With this information you can then cut your CNAME records in your DNS service of choice in this arrangement:

```
sas-cas-server-default-backup-dc-pisces.viya.acme.com =>
a7cc66edbcald402c8c72ea7da3543d7-1688256446.us-east-1.elb.amazonaws.com
sas-cas-server-default-bin-pisces.viya.acme.com =>
a5a90408edfc246bba051f830d6e58fe-2016925591.us-east-1.elb.amazonaws.com
sas-cas-server-default-controller-dc-pisces.viya.acme.com =>
a14ca0a2c8009415891c05053ee36d83-222471472.us-east-1.elb.amazonaws.com
...
```

Configure Model Access

Configure Access to Models

Note: SAS Model Manager is only available as part of the SAS Model Manager, SAS Visual Data Science, SAS Visual Data Science Decisioning, and SAS Visual Machine Learning offerings.

In order to import models into SAS Model Manager and register models from Model Studio, SAS Visual Analytics, and SAS Studio into the common model repository, as well as add a model from the common model repository into a decision flow, users must have the appropriate access permissions. For more information, see [“Access to Models” in SAS Viya: Models Administration](#).

Configure Access to Analytic Store Model Files

Note: SAS Model Manager is only available as part of the SAS Model Manager, SAS Visual Data Science, SAS Visual Data Science Decisioning, and SAS Visual Machine Learning offerings. SAS Intelligent Decisioning is only available as part of the SAS Visual Data Science Decisioning and SAS Intelligent Decisioning offerings. If your deployment does not include any of these offerings, you should skip this section.

CAUTION

To use analytic store models, you must have completed the tasks described in the README file, “SAS Micro Analytic Service ASTORE Configuration”. The README file is located at `$deploy/sas-bases/examples/sas-microanalytic-score/astores/README.md` (for Markdown) and `$deploy/sas-bases/docs/sas_micro_analytic_service_astore_configuration.htm` (for HTML).

In order to publish analytic store models from SAS Model Manager, Model Studio, and SAS Intelligent Decisioning to the SAS Micro Analytic Service destination, see [“Accessing Analytic Store Model Files” in SAS Viya: Models Administration](#).

Configure SAS Data Quality

Configure the Quality Knowledge Base

For an overview of SAS Data Quality, see [“Overview” in SAS Viya: QKB Management](#). SAS Data Quality relies on a collection of rules and reference data called a SAS Quality Knowledge Base (QKB).

Your deployment includes the latest version of the QKB for Contact Information. It is configured to use the `English, United States` locale by default. To deploy a custom QKB into CAS, you should first create a QKB Archive and import it into CAS using SAS Environment Manager. For information about these tasks, see [“Create a QKB Archive \(QARC\) File” in SAS Viya: QKB Management](#).

If you are upgrading from SAS Viya 3.5, you must manually import any older or custom QKBs into this version. For instructions to import the QKB into SAS Studio, see [“How To \(Kubernetes\)” in SAS Viya: QKB Management](#) and [“Set the Default QKB and the Default Locale” in SAS Viya: QKB Management](#). To import a QKB into CAS, create a QARC archive and use the SAS Environment Manager; for details, see [“Create a QKB Archive \(QARC\) File” in SAS Viya: QKB Management](#).

Configure SAS Data Science Programming

Additional Software Associated with SAS Data Science Programming

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS Viya 3.5 software order is associated with deploying SAS Data Science Programming. The software order contains SAS Embedded Process for Hadoop and SAS Embedded Process for Teradata on Linux. The Software Order Email (SOE) specifies the location of the deployment documentation: [SAS Embedded Process: Deployment Guide](#). You should deploy this order only if you are using SAS

in-Database Technologies for Hadoop or SAS in-Database Technologies for Teradata.

A SAS 9.4 order is associated with deploying SAS Data Science Programming. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOEs for the associated orders do not refer to SAS Data Science Programming explicitly.

Configure SAS Event Stream Processing

In order to start using SAS Event Stream Processing, you must first use SAS Event Stream Processing Studio to launch a project. This causes the ESP Operator to start an ESP Server pod, which then runs the project.

For more information, see [“Running Event Stream Processing Projects” in SAS Event Stream Processing: Overview](#).

Configure SAS for Microsoft 365

For the information to complete the configuration of this product, including required security configuration, see [“Steps for SAS Administrator” in SAS for Microsoft 365: User’s Guide](#).

Configure SAS Intelligent Decisioning

Note: SAS Intelligent Decisioning is only available as part of the SAS Visual Data Science Decisioning and SAS Intelligent Decisioning offerings. If your deployment does not include any of these offerings, you should skip this section.

You must perform post-installation tasks such as configuring access to analytic store model files and setting configuration properties for SAS Intelligent Decisioning. For more information, see [SAS Intelligent Decisioning: Administrator’s Guide](#).

Configure SAS Model Manager

Note: SAS Model Manager is only available as part of the SAS Model Manager, SAS Visual Data Science, SAS Visual Data Science Decisioning, and SAS Visual Machine Learning offerings. If your deployment does not include any of these offerings, you should skip this section.

You must perform post-installation tasks such as configuring access to analytic store model files and a workflow client user account for use with SAS Model Manager. For more information, see [SAS Model Manager: Administrator's Guide](#).

Configure SAS Risk Modeling

To complete the configuration of this product, you must perform post-installation tasks such as working with user groups and creating a destination for publishing a model. For more information, see [SAS Risk Modeling: Administrator's Guide](#) (access key required).

Configure SAS Visual Analytics

Additional Software Associated with SAS Visual Analytics

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS 9.4 order is associated with deploying SAS Visual Analytics. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOE for the associated order does not refer to SAS Visual Analytics explicitly.

Configure SAS Visual Data Science

Additional Software Associated with SAS Visual Data Science

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS 9.4 order is associated with deploying SAS Visual Data Science. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOE for the associated order does not refer to SAS Visual Data Science explicitly.

Configure SAS Visual Data Science Decisioning

Additional Software Associated with SAS Visual Data Science Decisioning

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS Viya 3.5 software order is associated with deploying SAS Visual Data Science Decisioning. The software order contains SAS Embedded Process for Hadoop and SAS Embedded Process for Teradata on Linux. The Software Order Email (SOE) specifies the location of the deployment documentation: [SAS Embedded Process: Deployment Guide](#). You should deploy this order only if you are using SAS in-Database Technologies for Hadoop or SAS in-Database Technologies for Teradata.

A SAS 9.4 order is associated with deploying SAS Visual Data Science Decisioning. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOEs for the associated orders do not refer to SAS Visual Data Science Decisioning explicitly.

Configure SAS Visual Machine Learning

Additional Software Associated with SAS Visual Machine Learning

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS 9.4 order is associated with deploying SAS Visual Machine Learning. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOE for the associated order does not refer to SAS Visual Machine Learning explicitly.

Configure SAS Visual Statistics

Additional Software Associated with SAS Visual Statistics

If you want to access Cloud Analytic Services (CAS) with R or Python, you must download and deploy SWAT packages for them. Those packages are available at the following locations:

- [SWAT for Python](#)
- [SWAT for R](#)

A SAS 9.4 order is associated with deploying SAS Visual Statistics. The software order contains SAS Visual Analytics Add-In for Office. The SOE specifies the location of the deployment documentation.

Note: The SOE for the associated order does not refer to SAS Visual Statistics explicitly.

Configure SAS/CONNECT Spawner

If you are performing a new deployment and added a reference to the enable-spawned-servers.yaml example file to your base kustomization.yaml in order to spawn SAS/CONNECT servers in the spawner pod, you must make a change to the configuration in SAS Environment Manager.

- 1 Go to SAS Environment Manager.
- 2 Select **Configurations**, then **SAS/CONNECT Spawner**.
- 3 In the contents field for **sas.connect.spawner: startup_commands**, remove `- nolocallaunch` from the USERMODS line.
- 4 Click **Save**.

For more information about this task, see the “Allow the Ability to Spawn Servers within the Spawner Pod” section of the “Configure SAS/CONNECT Spawner in SAS Viya” README file, located at `$deploy/sas-bases/examples/sas-connect-spawner/README.md` (for Markdown format) and at `$deploy/sas-bases/docs/configure_sasconnect_spawner_in_sas_viya.htm` (for HTML format).

Validating the Deployment

<i>SAS Viya: Deployment Validation</i>	95
<i>SAS Viya and the SAS Operational Quality Tool</i>	95

SAS Viya: Deployment Validation

Validating your SAS Viya deployment is not a Kubernetes-based task and requires a SAS administrator. The procedures for validating products in your deployment are described in [SAS Viya: Deployment Validation](#).

SAS Viya and the SAS Operational Quality Tool

SAS Operational Qualification Tool (SAS OQ) helps qualify the use of SAS software in regulated industries. SAS OQ supports the qualification aspect of the essential migration, integration, and verification processes that customers must perform to validate the SAS deployment. SAS OQ helps to demonstrate that SAS is operational. For more information about SAS OQ, see [SAS Operational Qualification Tool: User's Guide](#).

Modifying the Deployment Configuration

Modify Existing Customizations in a Deployment 97

Modify Existing Customizations in a Deployment

After your software has been deployed, you might want to modify some of the customizations that you made in your initial deployment. For example, you might want to move from an SMP deployment of CAS to an MPP deployment. Or you might want to move from using TLS in front-door mode to using full-stack TLS.

Modifying your configuration does not require new software. To modify the customizations, simply repeat some of the steps that you performed in the initial deployment of your software:

IMPORTANT Performing these steps causes an outage while the software is re-deployed with the new configuration settings. Ensure that you plan for an outage before continuing with the steps..

- 1 Perform the modifications. Refer to the topic in [“Common Customizations” on page 29](#) or the README file that describes the changes that should be made. The README indexes, located at `$deploy/sas-bases/README.md` for README files in Markdown language or `$deploy/sas-bases/docs/index.htm` for README files in HTML, can help direct you to the appropriate README.
- 2 Redeploy the software.
 - If you are deploying with the SAS Viya Deployment Operator, first modify the SASDeployment custom resource with the command provided at [“Run the](#)

[create sas-deployment-cr Command](#) on page 65. Then run the deployment command described at [“Command and Output” on page 69](#).

- If you are performing a manual deployment, run the series of commands that are described at [“Manual Deployment” on page 71](#).

Uninstalling

<i>Uninstall with the SAS Viya Deployment Operator</i>	99
Remove the SAS Prepull and SAS Workload Orchestrator	
ClusterRoles and ClusterRoleBindings	99
Uninstall a SAS Viya Deployment from a Namespace Only	100
Remove All SAS Viya Deployments in the Cluster and the SAS	
Viya Deployment Operator	101
Remove the SAS Viya Deployment Operator Only	101
Remove Additional Resources	102
Validate the Removal	103
<i>Uninstall Manually</i>	103
Remove the SAS Prepull and SAS Workload Orchestrator	
ClusterRoles and ClusterRoleBindings	103
Remove Service Account Links and SCCs	104
Remove SAS Viya	105
Remove CRDs	105

Uninstall with the SAS Viya Deployment Operator

Remove the SAS Prepull and SAS Workload Orchestrator ClusterRoles and ClusterRoleBindings

Before you begin to uninstall your SAS Viya software, whether from a single namespace or from an entire cluster, perform the following tasks to account for some special cluster-wide permissions.

- 1 If you enabled the Image Staging Node List Option in your deployment, follow the instructions in the “Disable the Node List Option” section of the “SAS Image

Staging Configuration Options” README located at `$deploy/sas-bases/examples/sas-prepull/README.md` (for Markdown format) and `$deploy/sas-bases/docs/sas_image_staging_configuration_options.htm` (for HTML format).

- 2 If your deployment includes Workload Management and Cluster Role was enabled for it, follow the instructions in the “Disable the Cluster Role” section of the “Cluster Privileges for SAS Workload Orchestrator Service” README located at `$deploy/sas-bases/examples/sas-workload-orchestrator/configure/README.md` (for Markdown format) and `$deploy/sas-bases/docs/configuration_settings_for_sas_workload_orchestrator_service.htm` (for HTML format).

Uninstall a SAS Viya Deployment from a Namespace Only

Remove SAS Viya from a Namespace

When the operator deploys SAS Viya, it assigns an ownerReference from the custom resource to every namespace-scoped resource deployed by the operator. When the custom resource is deleted, Kubernetes deletes anything that was owned by that resource, thereby uninstalling SAS Viya. However, some resources in a SAS Viya deployment are cluster-scoped (such as CRDs, ClusterRoles, and ClusterRoleBindings). These are shared by all SAS Viya deployments in the cluster, so they are not considered “owned” by any one custom resource. Therefore, they are not deleted when the custom resource is removed for a namespace.

To uninstall a deployment of SAS Viya from a namespace without uninstalling the operator, remove the SASDeployment custom resource from that namespace. If you kept your `$deploy-sasdeployment.yaml` file, run the following command:

```
kubectl -n name-of-namespace delete -f $deploy-sasdeployment.yaml
```

If you did not keep the `$deploy-sasdeployment.yaml` file, run the following command:

```
kubectl -n name-of-namespace delete sasdeployments SASdeployment-custom-resource-name
```

When the command has finished running, SAS Viya is no longer running in the namespace, but the SAS Viya Deployment Operator is still running in the cluster. If you have removed the last SAS Viya deployment from the cluster and want to remove any remaining cluster-scoped resources, see [“Remove the Operator CustomResourceDefinition” on page 102](#).

Remove Service Account Links

Note: If your deployment is not on Red Hat Openshift, skip this section.

For Red Hat OpenShift systems, the namespace's service accounts must be removed from the assigned SCCs. To find the namespace's service accounts that are linked to SCCs, run the following code to generate the commands to review and execute:

```
SAS_NS=name-of-namespace
for oc_scc in $(oc get clusterrolebinding | grep
"system:openshift:scc:" | awk '{ print $1 }'); do
    crb_name=$(oc get clusterrolebinding $oc_scc -o json | jq -r
'.roleRef.name')
    sa_names=$(oc get clusterrolebinding $oc_scc -o json | jq -r
'.subjects[] | select(.namespace == \"$SAS_NS\") | .name')
    for sa_name in $sa_names; do
        if [[ ! -z $sa_name ]]; then
            echo "oc -n $SAS_NS adm policy remove-scc-from-user $
{crb_name##*:} -z $sa_name"
        fi
    done
done
```

Run each command that is produced as output from the code.

Remove All SAS Viya Deployments in the Cluster and the SAS Viya Deployment Operator

To uninstall all the SAS Viya deployments in a cluster, repeat the instructions in [“Remove SAS Viya from a Namespace” on page 100](#) for each namespace in the cluster that contains a SAS Viya deployment. After all the SAS Viya deployments have been removed, run the following command to remove the SAS Viya Deployment Operator:

```
kustomize build . | kubectl -n name-of-deployment-operator-namespace delete -f -
```

After the operator has been removed, see [“Additional Clean-up of Cluster Resources” on page 102](#) to remove any cluster-scoped resources.

Remove the SAS Viya Deployment Operator Only

To remove SAS Viya Deployment Operator from the cluster but leave the instances of SAS Viya running in their namespaces, run the following commands:

```
kubectl delete serviceaccounts,roles,rolebindings,deployments,secrets \
-n name-of-deployment-operator-namespace \
--selector "app.kubernetes.io/name=sas-deployment-operator"

kubectl delete clusterrolebindings \
$(kubectl get clusterrolebindings \
--selector "app.kubernetes.io/name=sas-deployment-operator" \
-o jsonpath='{.items[?(@.subjects[0].namespace == "name-of-deployment-operator-namespace")].metadata.name}')
```

After the commands are completed, see [“Additional Clean-up of Cluster Resources” on page 102](#) for information to remove some remaining resources.

Remove Additional Resources

Additional Clean-up of Cluster Resources

After all of the SAS Viya Deployment Operator instances are removed from the cluster, some shared cluster resources must be removed.

```
kubectl delete clusterroles \
  --selector "app.kubernetes.io/name=sas-deployment-operator"
```

Remove the Operator CustomResourceDefinition

After all of the SASDeployment custom resources have been removed from the cluster, the operator CustomResourceDefinition must be removed.

```
kubectl delete crds \
  --selector "app.kubernetes.io/name=sas-deployment-operator"
```

Remove SCCs

Note: If your deployment is not on Red Hat Openshift, skip this section.

If you have removed the last namespace running SAS Viya in the cluster, then the SAS-specific SCCs can be removed. The following can be run to generate the commands to review and execute:

Note: Removing SCCs is a cluster-wide operation. Ensure that no namespaces with SAS Viya software exist in the cluster before removing SCCs. Otherwise, any remaining SAS Viya deployments in the cluster may cease to work as expected.

```
for sas_scc in $(oc get scc | grep sas | awk '{ print$1 }'); do echo
"oc delete scc $sas_scc"; done
for sas_scc in $(oc get scc | grep pgo | awk '{ print$1 }'); do echo
"oc delete scc $sas_scc"; done
```

Run each command that is produced as output from the code.

Validate the Removal

- 1 After you have removed the last namespace with SAS Viya software from the cluster, run the following commands:

```
kubectl get crd --selector "sas.com/admin=cluster-wide"
kubectl get crd --selector "sas.com/admin=cluster-api"
kubectl get clusterrole --selector "sas.com/admin=cluster-wide"
```

If you removed the cluster-wide CRDs, each command should return zero results. If you do get results, repeat the steps above to remove the CRDs.

- 2 If you are removing a deployment from OpenShift, run the following commands:

```
kubectl get scc | grep sas
kubectl get scc | grep pgo
kubectl get clusterrole | grep "system:openshift:scc:sas-"
```

If you removed the cluster-wide SCC settings, each command should return zero results. If you do get results, repeat the steps above to remove the SCCs or ClusterRoles.

Uninstall Manually

Remove the SAS Prepull and SAS Workload Orchestrator ClusterRoles and ClusterRoleBindings

Before you begin to uninstall your SAS Viya software, whether from a single namespace or from an entire cluster, perform the following tasks to account for some special cluster-wide permissions.

- 1 If you enabled the Image Staging Node List Option in your deployment, follow the instructions in the “Disable the Node List Option” section of the “SAS Image Staging Configuration Options” README located at `$deploy/sas-bases/examples/sas-prepull/README.md` (for Markdown format) and `$deploy/sas-bases/docs/sas_image_staging_configuration_options.htm` (for HTML format).
- 2 If your deployment includes Workload Management and Cluster Role was enabled for it, follow the instructions in the “Disable the Cluster Role” section of the “Cluster Privileges for SAS Workload Orchestrator Service” README located at `$deploy/sas-bases/examples/sas-workload-orchestrator/configure/README.md` (for Markdown format) and `$deploy/sas-bases/docs/`

`configuration_settings_for_sas_workload_orchestrator_service.htm` (for HTML format).

Remove Service Account Links and SCCs

Note: If your deployment is not on Red Hat OpenShift, skip this section.

For Red Hat OpenShift systems, the namespace's service accounts must be removed from the assigned SCCs.

- 1 To find the namespace's service accounts that are linked to SCCs, run the following to generate the commands to review and run:

```
SAS_NS=name-of-namespace
for oc_scc in $(oc get clusterrolebinding | grep
"system:openshift:scc:" | awk '{ print $1 }'); do
    crb_name=$(oc get clusterrolebinding $oc_scc -o json | jq -r
'.roleRef.name')
    sa_names=$(oc get clusterrolebinding $oc_scc -o json | jq -r
'.subjects[] | select(.namespace == \"$SAS_NS\") | .name')
    for sa_name in $sa_names; do
        if [[ ! -z $sa_name ]]; then
            echo "oc -n $SAS_NS adm policy remove-scc-from-user $
{crb_name##*:} -z $sa_name"
        fi
    done
done
```

Run each command that is produced as output from the code.

- 2 If you have removed the last namespace running SAS Viya in the cluster, then the SAS-specific SCCs can be removed. The following can be run to generate the commands to review and run:

Note: Removing SCCs is a cluster-wide operation. Ensure that no namespaces with SAS Viya software exist in the cluster before removing SCCs. Otherwise, any remaining SAS Viya deployments in the cluster may cease to work as expected.

```
for sas_scc in $(oc get scc | grep sas | awk '{ print $1 }'); do echo
"oc delete scc $sas_scc"; done
for sas_scc in $(oc get scc | grep pgo | awk '{ print $1 }'); do echo
"oc delete scc $sas_scc"; done
```

Run each command that is produced as output from the code.

- 3 Run the following commands:

```
kubectl get scc | grep sas
kubectl get scc | grep pgo
kubectl get clusterrole | grep "system:openshift:scc:sas-"
```

If you removed the cluster-wide SCC settings, each command should return zero results. If you do get results, repeat the steps above to remove the SCCs or ClusterRoles.

Remove SAS Viya

Because your SAS Viya software is required to be deployed in a dedicated namespace, uninstall your deployment by deleting that namespace. As an administrator with cluster permissions, run the following command:

```
kubectl delete namespace name-of-namespace
```

When the namespace is deleted, your SAS Viya deployment has been uninstalled.

Note: If you have deployed SAS Viya on multiple namespaces, repeat the command on each namespace to remove SAS Viya from the cluster.

Remove CRDs

If you are removing the last namespace with a SAS Viya deployment from the cluster, the SAS custom resource definitions (CRDs) must be removed as well.

Note: Removing CRDs is a cluster-wide operation. Ensure that no namespaces with SAS Viya software exist in the cluster before removing CRDs. Otherwise, any remaining SAS Viya deployments in the cluster may cease to work as expected.

- 1 As a user with rights to list CRDs, run the following two commands to determine if any SAS CRDs still exist:

```
kubectl get crd --selector "sas.com/admin=cluster-wide"
```

```
kubectl get crd --selector "sas.com/admin=cluster-api"
```

- 2 If either of those commands return objects, run the following two commands as a user with rights to remove CRDs:

```
kubectl delete crd --selector "sas.com/admin=cluster-wide"
```

```
kubectl delete crd --selector "sas.com/admin=cluster-api"
```

- 3 Run the following commands:

```
kubectl get crd --selector "sas.com/admin=cluster-wide"
```

```
kubectl get crd --selector "sas.com/admin=cluster-api"
```

```
kubectl get clusterrole --selector "sas.com/admin=cluster-wide"
```

If you removed the cluster-wide CRDs, each command should return zero results. If you do get results, repeat the steps above to remove the CRDs.

Appendix 1

(Optional) Using a Mirror Registry

Create a Mirror Registry	107
About SAS Mirror Manager	108
System Requirements	108
Download Required Files	109
Create a Basic Mirror Registry	111
Required Flags	111
Optional Flags	112
Deploy the Software from the Registry	113
Create and Populate a Mirror Registry in Microsoft Azure	114
Additional Requirements	114
Configure Authentication for Azure Container Registry	114
Create and Populate a Mirror Registry in Amazon ECR	115
Additional Requirements	115
Using SAS Mirror Manager on AWS	115
Create and Populate a Mirror Registry in Google Cloud Platform	116
Additional Requirements	116
Create the Registry	117
Grant Permissions to the SAS Mirror Manager User	117
Using SAS Mirror Manager on GCP	117
Create a Mirror Registry at a Dark Site	118

Create a Mirror Registry

SAS Mirror Manager is a command-line utility that pulls SAS container images from the SAS Container Registry and pushes the images to your container registry, which can be either on-premises or in the cloud.

A mirror registry is optional, but it supports specific use cases, such as sites with limited internet access or those with a requirement to perform pre-deployment security scanning of SAS packages. Setting up the mirror requires access to the internet. However, after mirroring has been completed, the deployment can proceed at sites that lack internet access.

About SAS Mirror Manager

In addition to enabling deployment from a pre-populated mirror registry, SAS Mirror Manager supports other use cases. For example, you can use it to get a detailed list of the available SAS Viya stable and cadence releases in the SAS Container Registry. SAS Mirror Manager can help you determine whether differences exist between a SAS Viya repository and the current contents of the SAS Container Registry.

Using the SAS Mirror Manager list functionality, you can run checks before creating a mirror registry for your organization. Here are some examples of those checks:

To display a list of all the available software cadences and versions for a SAS software order:

```
mirrormgr list remote cadences --deployment-data SASViyaV4_order-  
number_certs.zip
```

To display a list of all the available software versions and releases:

```
mirrormgr list remote cadence releases --deployment-data  
SASViyaV4_order-number_certs.zip
```

To determine the required disk space for the available images:

```
mirrormgr list remote repos size --latest --deployment-data  
SASViyaV4_order-number_certs.zip
```

System Requirements

The number of containers and the size of the images that you obtain from SAS depend on the products in your software order. SAS Mirror Manager downloads each image locally and then uploads it to the destination registry. The images that are downloaded require approximately 30–50 GB of disk space on the machine on which SAS Mirror Manager is running. You can use the `--remove-after-upload` option with the `mirrormgr mirror registry` command to remove container images from the local cache after they have been uploaded to the destination registry. However, make sure that you have enough space to store all of the concurrent downloads.

You can run SAS Mirror Manager on 64-bit Linux, Windows, or macOS. By default, SAS Mirror Manager performs the number of concurrent downloads that corresponds to the number of available CPUs. You can override this setting with the `--workers` option and a value that defines the number of concurrent threads to be used for downloads. For more information, see [“Required Flags” on page 111](#).

If you plan to use the SAS Viya Deployment Operator with SAS Mirror Manager, perform the steps to download SAS Mirror Manager and create the mirror before deploying the operator. When you deploy the software for the operator, download the certificates again. For more information, see [“Deploy the SAS Viya Deployment Operator” on page 7](#).

SAS Mirror Manager does not support mirroring of Harbor versions 1.9.x and 1.10.x because of a Harbor limitation. Harbor versions earlier than 1.9.0 are supported, and Harbor version 2.0 and later are supported.

Before you run SAS Mirror Manager, perform the following steps:

- Configure your firewall to enable outbound connections to `ses.sas.download`, `ses.sas.com`, `crbwp.sas.com`, `bwp2.ses.sas.download`, and `cr.sas.com` on port 443.

- Verify network bandwidth.

SAS Mirror Manager requires a high-speed network environment. A lack of sufficient bandwidth increases the amount of time that is required to create the mirror registry and can lead to time-out errors when communicating with the destination registry.

- Determine the authentication mechanism for the target registry.

If you are using a cloud registry provider, use the corresponding CLI to access the user name and password for your cloud registry. The user account that is specified in the authentication must have permission to push images to the cloud registry.

Note: SAS Mirror Manager does not support mirroring container images to a Red Hat OpenShift Platform registry at this time.

See the following documentation for more information about the target registry:

- Azure Container Registry: [ACR documentation](#)
- Amazon Elastic Container Registry (Amazon ECR): [Amazon ECR documentation](#)
- Google Cloud Platform Container Registry: [GCR documentation](#)

In a multi-tenant deployment, all tenant onboarding and offboarding YAML files must include the transformer file `site-config/mirror.yaml`. For more information, see the README file at `$deploy/sas-bases/examples/sas-tenant-job/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/onboard_or_offboard_tenants_using_the_tenant_job.htm` (for HTML format).

Download Required Files

To create a mirror registry with SAS Mirror Manager, start by obtaining the software.

When you order SAS software, SAS sends a Software Order Email (SOE) to your business or organization. Your SOE includes information about the software order, including a link to `my.sas.com`.

- 1 Create a directory on your kubectl machine or on a machine that can be reached by your kubectl machine:

```
mkdir directory-name
```

SAS recommends that you name the directory `deploy`, but you should use a name that is meaningful to you. The directory is referred to as `$deploy` in this guide.

- 2 Click the **Get Started** link that is provided in your SOE and log in to my.sas.com.
- 3 On the My SAS web page that opens, expand the information for the order that is listed by clicking the down arrow.
- 4 In the pane that opens, examine the order information. The version indicates the release cadence and the version of SAS Viya software you are about to deploy. If you want to deploy a different version, select the cadence and release from the **SAS Viya Version** list.

- 5 Create a new directory parallel to the `$deploy` directory in which to store your license and *-certs.zip files.

```
mkdir directory-name
```

You should use a name that is meaningful to you. The directory is referred to as `$license` in this guide.

- 6 Retrieve the certificates and license to deploy your software.
Click **Download Certificates** and download the *-certs.zip file. Place the ZIP file in the `$license` directory.
- 7 Click **Download License Only** to download the license file, identified by the .jwt extension. Save the JWT file in the same directory.
- 8 Click **Download Deployment Assets** to download the TGZ archive of the files that are required to deploy your software.

Note: For some products, the **Download Deployment Assets** button is not available because these assets are not required.

- 9 Save the TGZ file from the my.sas.com page to the directory that you created in step 1.
- 10 Extract the files from the TGZ file in the same directory:

```
tar xvfz file-name.tgz
```

- 11 Scroll down to the section of the page that is labeled **SAS Mirror Manager**. Click the **Download Now** link to download the SAS Mirror Manager package to the machine where you want to create your mirror registry.

If you use Internet Explorer to download the Linux or Macintosh version, save the file in TGZ format instead of GZ format.

Note: Be sure to download the SAS Mirror Manager software from the [SAS Mirror Manager download site](#). If you have a version of SAS Mirror Manager that you used with an earlier version of SAS Viya, it is not compatible with the current architecture of the SAS Container Registry.

- 12 Uncompress the SAS Mirror Manager file, `mirrormgr-operating-system.tgz`, in your `$deploy` directory.
- 13 (Optional) SAS recommends saving the SOE in the same directory.

Create a Basic Mirror Registry

The following command creates and populates a mirror registry:

```
mirrormgr mirror registry \
  --destination myregistry.mydomain.com \
  --username myregistryuser \
  --password myregistrypassword \
  --deployment-data ~path-to-certs-zip-file
```

The `mirror registry` command performs the following tasks:

- Pulls images from the SAS Container Registry.
- Pushes the images to the specified private registry.

The optional `--deployment-assets` flag is recommended. Without it, SAS Mirror Manager picks up the latest release of all supported cadence versions, and additional disk space is required to accommodate the software that is downloaded. For more information, see [“Optional Flags” on page 112](#).

If the target registry for the images requires an `imagePullSecret` for access to those images, you will need to create one. For more information, see [“Use ImagePullSecrets to Access the Mirror Registry” on page 31](#).

For more information about mirroring to specific cloud provider container registries, see:

- [“Create and Populate a Mirror Registry in Microsoft Azure” on page 114](#)
- [“Create and Populate a Mirror Registry in Amazon ECR” on page 115](#)
- [“Create and Populate a Mirror Registry in Google Cloud Platform” on page 116](#)

Required Flags

Here is the basic command syntax, which includes required flags:

```
mirrormgr mirror registry required-flags
```

--deployment-data *path-to-certs-zip-file*

specifies the full path to the `certs.zip` file that was downloaded to the `$deploy` directory. You downloaded the ZIP file from `my.sas.com`.

--destination *registry-location*

specifies the host name and repository path of the registry to which images are replicated. Here is an example:

```
registry.example.com/my-project
```

--password *destination-registry-password*

specifies the password that grants you access to the destination registry.

--username *destination-registry-user-name*

specifies the user name that grants you access to the destination registry.

Optional Flags

--cacert *path-to-CA-certificate*

specifies the CA certificate for communicating with the upstream repository server. The default is `$HOME/.config/mirrormgr/ca.pem`.

--cadence *cadence-version*

specifies the cadence (or frequency) at which you deploy new SAS software. It can be used with the `--release` flag. Here is an example of specifying a release with the *stable* cadence, version 2020.1.1:

```
--cadence stable-2020.1.1
```

Note: The `--cadence` flag is not required. However, it enables you to mirror the exact container image versions from which the accompanying deployment orchestration assets were generated. By including the cadence and release, you ensure that the mirror provides all the images that are required for a SAS Viya deployment. If you have downloaded deployment assets from `my.sas.com`, use the `--deployment-assets` flag instead of the `--cadence` flag.

--cert *path-to-auth-certificate*

specifies the certificate for authenticating with the upstream repository server. The default is `$HOME/.config/mirrormgr/cert.pem`.

-d , --debug

enables debug-level logging.

--deployment-assets

uses data from the Deployment Assets archive that you downloaded with your software order to determine the cadence name, version, and release values. Replaces the `--cadence` and `--release` flags.

--latest

mirrors only the latest versions of SAS content.

--log-file *path-to-log-file*

specifies the log file location. The default is `$HOME/.local/share/mirrormgr/mirrormgr.log`.

-p , --path *path-to-repositories*

specifies the path where repositories are downloaded. The default is `$HOME/sas_repos`.

--platform *platform-filter*

specifies a filter for identifying the repositories to mirror, by platform. Here is an example of a platform filter:

```
--platform x64-oci-linux-2
```

--release *release-cadence-timestamp-ID*

specifies the software release that matches the deployment orchestration tools that are used in your deployment assets file (in TGZ format). The release includes the timestamp when the specified cadence version was last updated.

If the `--release` flag is used, the `--cadence` flag must also be used. If the `--release` flag is not used, SAS Mirror Manager mirrors the latest release of a cadence version.

Note: You can derive the cadence, version, and release from the file name of the deployment assets in the TGZ file that you download from my.sas.com. Here is an example file name:

```
SASViyaV4_09S468_stable-2020.1.1_20201016.1594946240211_deploymentAssets_2020-07-17T022620.tgz
```

In this example, `stable-2020.1.1` corresponds to the cadence and version, and the release includes a release year, month, and day plus an ID, such as `20201016.1594946240211`. However, if you have downloaded deployment assets, you can use the `--deployment-assets` flag to pass in the cadence and release automatically.

--remove-after-upload

removes container images from the local cache after they have been uploaded to a remote registry.

-r , --repo

specifies a list of repositories to mirror. If not specified, all entitled repositories are mirrored. To obtain a full list of repositories, run this command:

```
mirrormgr list remote repos
```

--url URL

specifies the base URL of upstream repositories. The default is `https://ses.sas.download/ses`.

--workers number

specifies the number of download threads to use. By default, the value is the same as the number of cores on the local machine.

A complete list of flags used in the command is available in the Help for SAS Mirror Manager. Access the Help using `-h` or `--help`.

Deploy the Software from the Registry

When you have verified that the SAS images to which you are entitled have been added to the specified registry, you are ready to deploy SAS Viya. Your next steps depend on your deployment method:

- If you are performing a deployment using the SAS Viya Deployment Operator, follow the steps in [“Deploy the SAS Viya Deployment Operator” on page 7](#) in order to prepare your environment.
- If you are performing a manual deployment, follow the steps in [“Retrieve Required Files” on page 13](#) to download additional files that are required for the deployment. Then you can proceed to the [Installation on page 25](#) steps.

Create and Populate a Mirror Registry in Microsoft Azure

SAS Mirror Manager requires some additional configuration when the target container registry is hosted on Microsoft Azure.

Additional Requirements

Based on the average size of a SAS container deployment, a single deployment can be performed from a Standard-sized Azure container registry. However, if you intend to have a long-running registry that consumes SAS software updates over time, you should select a Premium size. For more information, see the [Microsoft Azure sizing documentation](#).

Configure Authentication for Azure Container Registry

Several methods of authenticating to the Azure Container Registry are available. You can see a full list in the Azure container authentication docs. You can use the Azure CLI to do the following:

- 1 Log in to your Microsoft Azure account in order to authenticate your client:

```
az login
```

- 2 Get an authorization token:

```
az acr login --name registry_name --expose-token
```

For *registry_name*, substitute the name of the destination container registry.

The output that is returned includes the access token and the login server, as in the following example:

```
docker login loginServer -u 00000000-0000-0000-0000-000000000000 -p accessToken
{
  "accessToken": "token-contents",
  "loginServer": "mirrormgr.azurecr.io"
}
```

- 3 Run SAS Mirror Manager with your credentials:

```
mirrormgr mirror registry --deployment-data SASViyaV4_order-
number_certs.zip
--latest --destination registry-name.mydomain.com --username
00000000-0000-0000-0000-000000000000 --password access-token
```

For *registry-name.mydomain.com*, specify the `loginServer` that was returned from the previous command as the `--destination`.

For *access-token*, specify the Azure token that was returned previously.

Note: The user name must be 00000000-0000-0000-0000-000000000000.

Create and Populate a Mirror Registry in Amazon ECR

SAS Mirror Manager requires additional configuration when the target container registry is hosted on AWS.

Additional Requirements

In order to support Amazon ECR as the mirror destination, make sure you have fulfilled some additional requirements:

- The AWS CLI must be installed.
- The AWS CLI must be authenticated.
- The user account must have permission to create new Amazon ECR repositories. This permission is specified in the Identity and Access Management (IAM) section of the AWS console.

Using SAS Mirror Manager on AWS

The image repositories must be created in Amazon ECR before they are pushed. Take the following steps to use SAS Mirror Manager in an AWS environment:

- 1 Create or configure an AWS private registry with Amazon ECR.
Your AWS account has an associated default private Amazon ECR registry. For more information, see [Amazon ECR private registries](#).
- 2 Follow the steps in “[Download Required Files](#)” on page 109 to obtain the SAS Mirror Manager software.

- 3 Create any required repositories:

```
for repo in $(mirrormgr list target docker repos --deployment-data
SASViyaV4_order-number_certs.zip
--destination namespace) ; do
```

```
aws ecr describe-repositories --repository-names $repo --region
region || aws ecr create-repository --repository-name $repo --region
region
done
```

Note: The call to `describe-repositories` runs in order to determine whether the repositories exist. If they do not exist, an error message is returned to alert you about this condition. This error may be safely ignored.

For *namespace*, supply the name of the target namespace. For *region*, supply the name of the AWS region. Be sure to use the same region in all commands.

- 4 Create and populate the mirror based on the associated software order:

```
mirrormgr mirror registry -p ./sas_repos --deployment-data
SASViyaV4_order-number_certs.zip
--latest --destination registry-URL/namespace --username 'AWS'
--password $(aws ecr get-login-password --region region)
```

The same options that are described in “Optional Flags” on page 112 are also applicable to Amazon ECR.

If you encounter errors that report too many requests, you can reduce the number of worker threads. By default, the number of threads that is used is the same as the number of CPU cores on the machine. Use the `--workers` option to set the number of threads.

- 5 Proceed to “Deploy the Software from the Registry” on page 113 for additional deployment instructions.

Create and Populate a Mirror Registry in Google Cloud Platform

SAS Mirror Manager requires additional configuration when the target container registry is hosted in GCP.

Additional Requirements

In order to support Google Cloud Platform as the mirror destination, make sure you have fulfilled the following additional requirements:

- To create a mirror registry in GCP, the GCP SDK is required. For instructions to download and install the GCP SDK, see <https://cloud.google.com/sdk/docs/quickstart-linux>.
- In order to create a mirror registry in GCP using SAS Mirror Manager on a Windows system, you must have Microsoft PowerShell.

Create the Registry

Create a container registry in your GCP account before you run SAS Mirror Manager. Follow these instructions:https://cloud.google.com/container-registry/docs/quickstart?hl=en_US.

Grant Permissions to the SAS Mirror Manager User

You must explicitly grant two storage-related permissions that enable SAS Mirror Manager to access your GCP environment.

To change required storage permissions:

- 1 Log in to your Google Cloud Platform account, and select **Cloud Storage** from the side panel.
- 2 Expand **artifacts.\$PROJECT.appspot.com** and select **permissions**.
- 3 In the **Add Members** field, enter the user name that you want to use to run SAS Mirror Manager commands. This account can be your user account or a service account.
- 4 Grant the following permissions to the account that you specified:
 - Storage Object Creator
 - Storage Object Viewer

Using SAS Mirror Manager on GCP

The command to run SAS Mirror Manager is slightly different in GCP environments. Use the `gcloud` command to fetch your access token. Here is an example:

```
mirror registry --deployment-data SASViyaV4_order-number_certs.zip --
latest --destination gcr.io/$GCP_PROJECT/$NAMESPACE --username
oauth2accesstoken --password $(gcloud auth print-access-token)
```

Note: The value for username is always `oauth2accesstoken`.

Create a Mirror Registry at a Dark Site

You can use SAS Mirror Manager to enable deployment of SAS Viya to an air-gapped network or to a network where systems are not allowed to communicate with hosts on the internet. SAS Mirror Manager enables you to download content to a local directory, from which you can copy the content to the isolated network. From there, you can upload SAS Viya content to an internal registry.

To deploy SAS Viya to a dark site:

- 1 Follow the steps in [“Download Required Files” on page 109](#) to obtain the SAS Viya software and entitlements and to download SAS Mirror Manager.
- 2 Mirror the software to a local directory.

```
mirrormgr mirror registry --path ./sas_repos --deployment-data
SASViyaV4_order-number_certs.zip
```

If you do not provide a destination to the `mirrormgr mirror registry` command, all files are downloaded to the directory that is specified by the `--path` flag. If you do not specify a path, `~/sas_repos` is used by default.

TIP Be sure to include any cadence version filters that you want to use in order to limit the software that is downloaded. For more information, see [“Optional Flags” on page 112](#).

- 3 Mirror the local directory to an internal registry.

```
mirrormgr mirror registry --deployment-data SASViyaV4_order-
number_certs.zip --path ./sas_repos --destination registry.example.com
--username registry-user-name --password registry-password --push-only
```

Specifying the `--destination` flag along with the `--push-only` flag mirrors the local copy to the specified registry without connecting to the SAS Entitlement Service or to the SAS Container Registry.

Appendix 2

SAS Viya Deployment Operator Fields and Messages

Overview	119
Fields in the SASDeployment Custom Resource	120
Primary Fields	120
User File Options	121
Secret Key Selector Fields	121
Repository Warehouse Options	122
Deployment User Content Options	122
Manage Updates	122
Environment Variables for the Operator Pod	123
Communications from the Operator	125
Overview	125
Status Fields	126
Deployment Condition Fields	126
Deployment Release Fields	127
Deployment Release Details Fields	127

Overview

This appendix contains lists of fields and variables that can be included in the files to configure the SAS Viya Deployment Operator or that are included in its status messages.

Fields in the SASDeployment Custom Resource

Primary Fields

The following fields can be added to the SASDeployment custom resource in the spec (specification) block.

Note: Apply the indentation rules from the preceding examples to new resources that you create

Table A2.1 Fields in the spec Block

Field	Description
caCertificate	Optional. Information about the certificate authority. See “User File Options” on page 121 for more information about the format.
clientCertificate	Optional. Information about the client certificate. See “User File Options” on page 121 for more information about the format.
cadenceName	Required. The name of the cadence to deploy.
cadenceRelease	Optional. The release of the cadence to deploy.
cadenceVersion	Required. The version of the cadence to deploy.
deploymentType	Optional. The type of deployment to use.
imageRegistry	Optional. The image registry to deploy from.
license	Optional. The license to apply to the deployment. For more information about the format, see “User File Options” on page 121
repositoryWarehouse	Optional. Information about the repository warehouse. See “Repository Warehouse Options” on page 122 for more information about the options.

Field	Description
userContent	Required. The user customizations. See “Deployment User Content Options” on page 122 for information about the options.

User File Options

A field that requires a user file can have one of three values:

Table A2.2 User File Options

Field	Description
content	A nested map.
secretKeyRef	A secret in the namespace of this SASDeployment. See “Secret Key Selector Fields” on page 121 for more information.
url	A <i>go-getter</i> URL from which to fetch content. See GitHub for more information about go-getters.

Secret Key Selector Fields

A field that requires a secret key selector has two parts:

Table A2.3 Secret Key Selector Fields

Field	Description
name	Required. The name of the secret.
key	Required. The name of the secret key.

Here is an example of the format:

```
spec:
...
  caCertificate:
    secretKeyRef:
      name: secret-name
      key: key-name
```

...

Repository Warehouse Options

The repository warehouse field has two options. For an example of the format, see [“Manage Updates” on page 122](#).

Table A2.4 Repository Warehouse Options

Field	Description
updatePolicy	Optional. The policy by which the operator automatically applies content changes to the repository warehouse. See “Manage Updates” on page 122 for details of the possible values.
url	Optional. The repository warehouse URL from which to deploy your SAS Viya software. The default is <code>https://ses.sas.download/ses</code> .

Deployment User Content Options

Deployment user content must use one of these two options:

Table A2.5 Deployment User Content Options

Field	Description
files	A nested map.
url	A go-getter URL from which to fetch content. See GitHub for more information about go-getters.

Manage Updates

The SAS Viya Deployment Operator can automatically download and apply updates to your SAS Viya deployment with the addition of the `updatePolicy` field in the custom resource.

Note: If your deployment includes an internal instance of PostgreSQL, do not add the `updatePolicy` field. Updating an internal instance of PostgreSQL currently requires manual steps that cannot be automated by the SAS Viya Deployment Operator.

```
...
spec:
  cadenceName: "LTS"
  cadenceVersion: "2020.1"
  cadenceRelease: "20200817.1597696671121"
  repositoryWarehouse:
  updatePolicy: {{ VALUE }}
```

Replace the `{{ VALUE }}` variable with one of the following values:

Never

The operator performs no updates. This is the default. Alternatively, if you do not want the operator to perform updates, you can omit the `updatePolicy` field entirely.

Releases

The operator automatically applies any `cadenceRelease` updates within the selected `cadenceName` and `cadenceVersion`. This setting restricts the updates to software with two decimal places in their version numbers. For example, the operator would update the deployment with version 2020.1.5, but not with version 2020.2.

Updates are checked once every 24 hours by a cronjob that is created in the SAS Viya namespace at deployment time. The exact start time is randomized with each deployment in an attempt to keep all of the deployments in a given time zone from updating at the same time. The schedule can be modified for all SAS Viya deployments managed by an operator by setting the `AUTOUPDATE_SCHEDULE` environment variable on the SAS Deployment Operator's Deployment resource. The schedule can also be modified for a specific deployment by setting the `environment.orchestration.sas.com/AUTOUPDATE_SCHEDULE` annotation on the SASDeployment Custom Resource for that deployment.

Environment Variables for the Operator Pod

The following variable is required in the pod specification of the file that deploys the operator.

SERVICE_ACCOUNT_NAME

Specifies the name of the service account under which the operator is running. This value is typically provided by the Downward API in Kubernetes.

The following variables are optional in the pod specification of the file that deploys the operator.

AUTOUPDATE_SCHEDULE

Specifies the schedule for the automatic update job, in CRON format. The default is daily. SAS recommends not setting this schedule to more often than once a day.

DEBUG_LOG_LOCATION

Enables debug logging to a file at the specified path. The directory that houses the file must exist and be writable. The special values `stdout` and `stderr` enable debug logging to standard output and standard error, respectively.

DISABLE_APPLY

Disables the application of the results of generation when the value is `true`. When combined with `DISABLE_CLEANUP`, this variable enables the user to test the operator results without making any changes to their cluster.

DISABLE_AUTO_RBAC

Disables the operator's attempts to ensure that the service account is running and has the appropriate permissions to deploy the content of the SASDeployment. When the variable is set to `true`, the Kubernetes cluster administrator is responsible for ensuring that the service account has the appropriate permissions. The operator creates the `$WORK_DIR/permissions` Kustomize base for the administrator to inspect or to manually apply.

DISABLE_CLEANUP

Disables cleanup of the work directory when the variable is set to `true`. The operator writes temporary files to the work directory. Users can mount persistent storage to this location and set this variable to `true` to test operator results.

DISABLE_LEADER_ELECTION

Disables leader election when the variable is set to `true`. The user must ensure that no more than one instance of the operator is managing the same namespace.

DISABLE_METRICS_SERVICE

Disables the metrics service when the variable is set to `true`. Otherwise, the operator starts a metrics endpoint and creates a Service resource in Kubernetes for that endpoint.

KUBECONFIG

When the operator is running outside of a Kubernetes cluster, this variable can be set to identify and provide access to a Kubernetes cluster. By default, the `~/.kube/config` file is used.

MINIMUM_UPDATE_INTERVAL

This variable is deprecated with version Stable 2020.1.1. While this variable is still supported, you should use `AUTOUPDATE_SCHEDULE` instead. See [“AUTOUPDATE_SCHEDULE” on page 124](#) for more information.

SAS_UPDATE_CHECK_INTERVAL

This variable is deprecated with version Stable 2020.1.1. While this variable is still supported, you should use `AUTOUPDATE_SCHEDULE` instead. See [“AUTOUPDATE_SCHEDULE” on page 124](#) for more information.

SERVICE_ACCOUNT_NAMESPACE

When the operator is running outside of a Kubernetes cluster, this variable can be set to specify the namespace of the service account under which the operator is running.

WATCH_NAMESPACE

Specifies the namespace to manage. If this variable is not set, or is set to an empty string, then the operator functions across the cluster and watches all namespaces.

WORK_DIRECTORY

Specifies an override for the work directory that is used by the operator.

Communications from the Operator

Overview

When the operator performs tasks, descriptions of actions and results are added to the status block in the SASDeployment custom resource. The following tables describe the format and content of the status block.

Here is an example of a status block for reference:

```

apiVersion: orchestration.sas.com/v1alpha1
kind: SASDeployment
...
status:
  conditions:
  - lastTransitionTime: "2020-10-06T13:19:24Z"
    message: Reconcile succeeded
    reason: ReconcileSucceeded
    status: "True"
    type: Ready
  displayName: Stable
  latest:
    releaseForCadence:
      cadenceName: stable
      cadenceRelease: "20201006.1601989747737"
      cadenceVersion: "2020"
      displayName: Stable
      matchesDeployedRelease: true
      supportEnd: "2030-03-15T04:00:00.000Z"
      supportLevel: SUPPORTED
    releaseForVersion:
      cadenceName: stable
      cadenceRelease: "20201006.1601989747737"
      cadenceVersion: "2020"
      displayName: Stable
      matchesDeployedRelease: true
      supportEnd: "2030-03-15T04:00:00.000Z"
      supportLevel: SUPPORTED
  messages:
  - ""
  - ""
  - |-
    role.rbac.authorization.k8s.io/sas-deployment-operator-mickey-test-
    a-sas-deployment created
  ...

```

Status Fields

The following fields can be added to the status block of the SASDeployment custom resource (referred to as *resource* in the table)..

Table A2.6 Deployment Status Fields

Field	Description
conditions	The current state of this SASDeployment. See “Deployment Condition Fields” on page 126 for more information.
displayName	The display name of the cadence for this SASDeployment.
messages	The list of messages from the last operation that was performed by the operator on this SASDeployment.
state	The state of this SASDeployment. Valid values are PENDING, RECONCILING, SUCCEEDED, or FAILED.
supportEnd	The date on which support for this SASDeployment ends.
supportLevel	The support level for this SASDeployment.

Deployment Condition Fields

The following fields are used to describe the condition of the SASDeployment custom resource.

Table A2.7 Deployment Condition Fields

Field	Description
type	The type of this condition.
status	The status of this condition.
lastTransitionTime	The timestamp that corresponds to the last status change of this condition.
reason	The reason for this condition's last transition.

Field	Description
message	The description of the last transition.

Deployment Release Fields

The following fields are used to describe the cadence release and version.

Table A2.8 *Deployment Release Fields*

Field	Description
releaseForVersion	The latest available release for the version of this SASDeployment.
releaseForCadence	The latest available release for the cadence of this SASDeployment.

For more information about either of these fields, see [“Deployment Release Details Fields” on page 127](#).

Deployment Release Details Fields

The following fields are used to provide details about the cadence release and version.

Table A2.9 *Deployment Release Details Fields*

Field	Description
cadenceName	The name of this available release.
cadenceRelease	The release of this available release.
cadenceVersion	The version of this available release.
displayName	The displayName of this available release.
matchesDeployedRelease	The value is <code>True</code> if this available release is currently deployed by this SASDeployment.
supportEnd	The date on which support for this available release ends.

Field	Description
supportLevel	The support level of this available release.

Appendix 3

PostgreSQL and Crunchy Data

<i>Overview</i>	129
<i>Available Pods</i>	130
sas-crunchy-data-pgadmin	130
sas-crunchy-data-pgbackrest-shared-repo	131
sas-crunchy-data-pgo-client	131
sas-crunchy-data-postgres	133
sas-crunchy-data-postgres-operator	136
sas-data-server-operator	137

Overview

SAS Viya includes an internal instance of PostgreSQL from Crunchy Data. If you choose to use the internal instance, descriptions of the pods that are included can help you plan your deployment.

This appendix describes the pods, whether they are deployed by default, and how to deploy them if they are not.

Note: For more information about internal and external instances of PostgreSQL, see [“Internal versus External PostgreSQL Instances” in System Requirements for SAS Viya](#).

Available Pods

sas-crunchy-data-pgadmin

This pod hosts the pgAdmin application, which is a web interface and administration tool for PostgreSQL databases. The `sas-crunchy-data-pgadmin` pod is not deployed by default. To deploy it, add `sas-bases/overlays/crunchydata_pgadmin` to the resources block of your base kustomization.yaml file. Here is an example:

```
...
resources:
...
- sas-bases/overlays/crunchydata_pgadmin
...
```

To retrieve the pgAdmin credentials:

To access pgAdmin:

- 1 Identify the pod that houses pgAdmin:

```
kubectl -n name-of-namespace -o wide get pods
```

The pgadmin pod name begins with `sas-crunchy-data-pgadmin`. Make note of the host name for the pgAdmin pod, which is used later if TLS enabled is in your deployment.

- 2 Set up port forwarding to port 5050 for the `sas-crunchy-data-pgadmin` pod:

```
kubectl -n name-of-namespace port-forward pgAdmin-pod --address 0.0.0.0
5050:5050
```

The value for `pgAdmin-pod` is the pod that was identified in Step 1.

When you run the port-forward command, the process continues running in the foreground. To run further kubectl commands, open a separate shell session. You can also run the port-forward command as a background process.

Note: If you are using Full-stack TLS or Front-door TLS, you can run the port-forward command from the Ingress host machine which is included as a Subject Alternative Name for the pgAdmin identity certificate.

- 3 Run the following commands to get the pgAdmin user ID and password.

- User ID:

```
kubectl -n name-of-namespace get secret sas-crunchy-data-pgadmin-
secrets -o 'go-template={{ .data.setupemail | base64decode }}'
```

The output of this command is the user ID in an email format.

- Password:

```
kubectl -n name-of-namespace get secret sas-crunchy-data-pgadmin-
secrets -o 'go-template={{ .data.setuppassword | base64decode }}'
```

The output of this command is the password.

- 4 Access pgAdmin in your browser: `http://port-forward-command-host-name:5050/pgadmin/login`. The value of *port-forward-command-host-name* is the name of the machine on which the port-forward command was run.

Note: If you are using Full-stack TLS or Front-door TLS, the protocol in the URL for accessing pgAdmin is https, rather than http.

Note: If you are using Full-stack TLS or Front-door TLS and receive an exception from the browser accessing the application, see “Extracting the SAS Viya generated root CA Certificate from a Deployment” in the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

sas-crunchy-data-pgbackrest-shared-repo

This pod provides the backup functionality of pgBackRest, which is a system for backing up and restoring PostgreSQL clusters. The pod runs pgBackRest backup operations as instructed by the Crunchy Data Operator. The backup enables the state of the PostgreSQL cluster to be restored later.

The `sas-crunchy-data-pgbackrest-shared-repo` pod is deployed by default when you deploy an internal instance of PostgreSQL.

The pod is invoked by the pgo client command `pgo backup`. For more information about the pgo client, see “[sas-crunchy-data-pgo-client](#)” on page 131.

sas-crunchy-data-pgo-client

This pod enables you to run the pgo client CLI against an installed PostgreSQL cluster. The PostgreSQL Operator Client, which is also known as pgo, is the most convenient way to interact with the PostgreSQL Operator. It provides convenient methods for creating, managing, and deleting PostgreSQL clusters through a series of simple commands. The pgo client interfaces with the API that is provided by the PostgreSQL Operator and can leverage the RBAC and TLS systems that are provided by the PostgreSQL Operator.

The `sas-crunchy-data-pgo-client` pod is not deployed by default. Instead, it can be invoked as necessary. To deploy it, add `sas-bases/examples/configure-postgres/internal/pgo-client` to the resources block of your base `kustomization.yaml` file.

```
...
resources:
```

```
...
- sas-bases/examples/configure-postgres/internal/pgo-client
...
```

To use the pgo client:

- 1 If a pgo-client job has already been created, delete it:

- a To determine whether you have a pre-existing job:

```
kubectl -n name-of-namespace get jobs
```

Check the output for the job name `sas-crunchy-data-pgo-client`.

- b If the pgo-client job is included in the output, delete it.

```
kubectl -n name-of-namespace delete job sas-crunchy-data-pgo-client
```

- 2 If you have not already copied the following content into the configMapGenerator block of your base kustomization.yaml file, copy it now.

```
configMapGenerator:
- name: pgo-command
  behavior: merge
  literals:
  - pgo-sub-command=pgo status sas-crunchy-data-postgres
  #- pgo-sub-command=pgo test sas-crunchy-data-postgres
  #- pgo-sub-command=pgo failover --query sas-crunchy-data-postgres
  #- pgo-sub-command=pgo show user sas-crunchy-data-postgres
  #- pgo-sub-command=yes|pgo restart sas-crunchy-data-postgres
  #- pgo-sub-command=yes|pgo scale --replica-count 1 sas-crunchy-data-postgres
  #- pgo-sub-command=yes|pgo update cluster --shutdown sas-crunchy-data-postgres
  #- pgo-sub-command=yes|pgo update cluster --startup sas-crunchy-data-postgres
  #- pgo-sub-command=pgo backup sas-crunchy-data-postgres --backup-type pgdump --database SharedServices --pvc-name sas-crunchy-data-postgres-dump-pvc
  #- pgo-sub-command=yes|pgo restore sas-crunchy-data-postgres --backup-type pgdump --pgdump-database SharedServices --backup-pvc sas-crunchy-data-postgres-dump-pvc
```

The literals list contains several common commands. The default command is `pgo test`. To run a different command in the list, uncomment its line by removing the number sign (#) and comment out the default command. Ensure that only one line is left uncommented. You can also add any pgo command to the list by using the appropriate format:

```
- pgo-sub-command=your-pgo-command
```

If you want to run more than one pgo command, separate them with a semicolon (;):

```
- pgo-sub-command=your-first-pgo-command ; your-second-pgo-command
```

For example, to check the status after you restart the server:

```
- pgo-sub-command=pgo test sas-crunchy-data-postgres ; pgo status sas-crunchy-data-postgres
```

- 3 To create a new pgo-client job, go to the `$deploy` directory and then run this command:


```
kustomize build . | kubectl apply -f -
```

- 4 Find the pgo-client pod and determine whether the status is *Complete*.

```
kubectl get pods -n name-of-namespace | grep pgo-client
```

If the status is not *Complete*, try again in a few minutes.

- 5 To see the results of the job, check the logs in the pod:

```
kubectl logs -n name-of-namespace pgo-client-pod
```

Specify the pod that was identified in Step 4 for *pgo-client-pod*.

sas-crunchy-data-postgres

The sas-crunchy-data-postgres pod is deployed by default when you deploy an internal instance of PostgreSQL. The sas-crunchy-data-postgres pod runs at deployment time.

database Container

This pod hosts the PostgreSQL RDBMS. This is used for internal management of PostgreSQL clusters, and it serves as the basis of your PostgreSQL deployments.

exporter Container

The exporter container in the sas-crunchy-data-postgres pod gathers metrics about PostgreSQL clusters that provide insight into cluster performance and activity.

pgbadger Container

The pgbadger container in the sas-crunchy-data-postgres pod hosts the pgBadger functionality, which parses PostgreSQL logs to generate reports. For SAS Viya, PostgreSQL logging is initially configured to be minimal, which might hinder the usefulness of this utility.

To access the pgBadger utility:

- 1 Find the pod that houses Crunchy PostgreSQL:

```
kubectl -n name-of-namespace get pods
```

The correct pod name begins with *sas-crunchy-data-postgres*.

- 2 Set up port forwarding for the sas-crunchy-data-postgres pod to the port 10000:

```
kubectl port-forward postgres-pod --address 0.0.0.0 10000:10000
```

Use the pod that was identified in Step 1 for *postgres-pod*.

- 3 Access pgBadger in your browser: `http://port-forward-command-host-name:10000`

The value of *port-forward-command-host-name* is the name of the machine on which the port-forward command was run.

pg_dump Functionality

The `sas-crunchy-data-postgres` pod contains the `pg_dump` functionality, which enables you to perform a backup of a database. For details to restore the backup later, see [“pg_restore Functionality” on page 135](#).

The pod is invoked by the pgo-client command `pgo backup`. To use the `pg_dump` functionality:

- 1 Construct the pgo-client command. Use the following command, replacing *database-name* with the name of the database that you want to back up. Use the name of a specific database rather than the name of the PostgreSQL cluster.

```
pgo backup sas-crunchy-data-postgres --backup-type pgdump --database
database-name --pvc-name sas-crunchy-data-postgres-dump-pvc
```

- 2 Follow the pgo-client steps, using your constructed pgo-client command as the `pgo-sub-command` value in the pgo-command `configMapGenerator`. For more information about the pgo-client, see [“sas-crunchy-data-pgo-client” on page 131](#).

Here is an example of the final `configMapGenerator` inside your base `kustomization.yaml`.

```
configMapGenerator:
- name: pgo-command
  behavior: merge
  literals:
  - pgo-sub-command=pgo backup sas-crunchy-data-postgres --backup-type pgdump
  --database SharedServices --pvc-name sas-crunchy-data-postgres-dump-pvc
```

- 3 Determine if the status of the `pg_dump` pod is *Complete*:

```
kubectl -n name-of-namespace get pods | grep pgdump
```

If the status is not *Complete*, try again in a few minutes.

Note: If you do not see any `pg_dump` pods, it is possible that the pgo-client pod failed. Perform [Step 4 on page 133](#) and [Step 5 on page 133](#) to check the logs of the pgo-client pod.

- 4 To see the results of the job, check the logs in the pod:

```
kubectl -n name-of-namespace logs pgdump-pod
```

For *pgdump-pod*, specify the pod that was identified in step 3.

pg_restore Functionality

The `sas-crunchy-data-postgres` pod contains the `pg_restore` functionality, which enables you to restore a backup of a database that was generated by the `pg_dump` functionality. The restore uses the latest backup from your PostgreSQL cluster. .

Note: For information about creating a backup, see [“pg_dump Functionality” on page 134](#)

The pod is invoked by the pgo-client command `pgo restore`. To use the `pg_restore` functionality:

- 1 Construct the pgo-client command. Use the following command, replacing *database-name* with the name of the database that you want to restore to. Because `pg_restore` does not create a new database for you, ensure that the destination database already exists in your PostgreSQL cluster.

```
yes|pgo restore sas-crunchy-data-postgres --backup-type pgdump --pgdump-
database database-name --backup-pvc sas-crunchy-data-postgres-dump-pvc
```

- 2 Follow the pgo-client steps, using your constructed pgo-client command as the `pgo-sub-command` value in the pgo-command `configMapGenerator`. For more information about the pgo-client, see [“sas-crunchy-data-pgo-client” on page 131](#).

Here is an example of the final `configMapGenerator` inside your base `kustomization.yaml`.

```
configMapGenerator:
- name: pgo-command
  behavior: merge
  literals:
  - pgo-sub-command=yes|pgo restore sas-crunchy-data-postgres --backup-type
pgdump --pgdump-database SharedServices --backup-pvc
sas-crunchy-data-postgres-dump-pvc
```

- 3 Determine if the status of the `pg_restore` pod is *Complete*:

```
kubectl -n name-of-namespace get pods | grep pgrestore
```

If the status is not *Complete*, try again in a few minutes.

Note: If you do not see any `pg_restore` pods, it is possible that the pgo-client pod failed. Perform [Step 4 on page 133](#) and [Step 5 on page 133](#) to check the logs of the pgo-client pod.

- 4 To see the results of the job, check the logs in the pod:

```
kubectl -n name-of-namespace logs pgrestore-pod
```

For *pgrestore-pod*, specify the pod that was identified in step 3.

pgBackRest Functionality

The `sas-crunchy-data-postgres` pod provides the restore functionality of `pgBackRest`, which is a system for backing up and restoring PostgreSQL clusters. It runs the `pgBackRest` restore operations, if a backup has already been performed.

The functionality is invoked by the `pgo` client command `pgo restore`. For more information about the `pgo` client, see [“sas-crunchy-data-pgo-client” on page 131](#).

sas-crunchy-data-postgres-operator

The `sas-crunchy-data-postgres-operator` pod is deployed by default when you deploy an internal instance of PostgreSQL. The `sas-crunchy-data-postgres-operator` is invoked by the `pgo-client`.

operator Container

This container runs the Crunchy PostgreSQL Operator, which controls PostgreSQL clusters. The Crunchy PostgreSQL Operator manages your PostgreSQL cluster resources and interacts with the various Crunchy Data pods to provide features to the user.

api-server Container

The `sas-crunchy-data-postgres-operator` pod hosts the Crunchy Data Operator API, which enables you to use the Crunchy Data Operator to query status or perform operations.

scheduler Container

The `sas-crunchy-data-postgres-operator` pod runs a scheduler for the Crunchy Data Operator, which enables repeatable tasks and aids in scheduling operations.

event Container

The `sas-crunchy-data-postgres-operator` pod runs an event manager for the Crunchy Data Operator, indicating activities within the PostgreSQL cluster.

sas-data-server-operator

This pod monitors the deployment for changes in the PostgreSQL cluster and handles configuration requirements for SAS Viya microservices.

The sas-data-server-operator pod is deployed by default when you deploy an internal instance of PostgreSQL.

The sas-data-server-operator pod runs at deployment time.

Appendix 4

Changing Deployment Methods

<i>Overview</i>	139
<i>Deploy the SAS Viya Deployment Operator</i>	139
<i>Create the \$license Directory</i>	140
<i>Create the SASDeployment Custom Resource</i>	140
<i>Deploy the Software</i>	140

Overview

This appendix describes how to use the SAS Viya Deployment Operator on a deployment that was manually deployed. It assumes that you have a working SAS Viya deployment that uses the general guidelines described in this document.

Deploy the SAS Viya Deployment Operator

If you have not already deployed the SAS Viya Deployment Operator, follow the instructions in [“Deploy the SAS Viya Deployment Operator” on page 7](#).

Create the \$license Directory

The SAS Viya Deployment Operator needs an explicit location for the license file and the zip file that contains the certificates for your software order. To create this directory, see step 6 in [“Retrieve Required Files” on page 13](#).

Create the SASDeployment Custom Resource

To create the custom resources required by the SAS Viya Deployment Operator, follow the instructions in [“Create the SASDeployment Custom Resource” on page 64](#).

When you reach the step to issue the `create sas-deployment-cr` command, you must enter the correct values for the `--cadence-*` fields. To find those values, run the following command:

```
kubectl get cm -o yaml | grep ' SAS_CADENCE'
```

Use the output of this command to determine the correct values for the `create sas-deployment-cr` command.

Note: If you have manually modified any of the resources since SAS Viya was last deployed, those changes are overwritten and replaced with their original values when this command is run.

Deploy the Software

Follow the instructions in [“Deployment Using the SAS Viya Deployment Operator” on page 69](#) to deploy the software using the SAS Viya Deployment Operator.

After you have confirmed that the operator is working as you expect, follow the instructions at [“Update to a New Version Using the Deployment Operator” in SAS Viya Operations: Updating Software](#) to update your software.

Appendix 5

Troubleshooting

<i>RequestEntityTooLarge Error Message</i>	141
<i>Error Checking Database Connection Error Message</i>	142
<i>Waiting for End of Discovery Error Message</i>	142
<i>Certframe Container Fails at init Stage</i>	143
<i>Crunchy PostgreSQL Operator etcd Error Message</i>	144
<i>Invalid Subject Name in Partner's Certificate Error Message</i>	145
<i>SAS Viya Deployment Operator Reconcile Failures</i>	146
<i>Performing Updates Produces Many Error Messages</i>	146
<i>Failure to Create Symbolic Link</i>	146
<i>Long Delay for SAS/CONNECT Server Sign-on from SAS Studio</i>	147

RequestEntityTooLarge Error Message

Error

When you are performing a deployment with the SAS Viya Deployment Operator and you attempt to apply the SASDeployment custom resource, the following error message is displayed:

```
Error from server (RequestEntityTooLarge)
```

Explanation

Applying the custom resource requires that it not exceed a size limit. The `RequestEntityTooLarge` error indicates that the custom resource exceeds that limit.

Resolution

When the custom resource is created, it consumes the content of the `$deploy` directory. To minimize the size of the custom resource, ensure that only the files that are needed are included in the `$deploy` directory. For example, `site.yaml` files from earlier manual deployments and old or duplicate `kustomization.yaml` files should be moved from the `$deploy` directory or deleted.

Error Checking Database Connection Error Message

Error

The following error message is displayed:

```
Error checking database connection: x509: certificate signed by unknown authority
```

Explanation

SAS Viya cannot connect securely with a database during deployment.

Resolution

For instructions to provide CA certificates to SAS Viya, see the "Incorporating Additional CA Certificates into the SAS Viya Deployment" section of the README file at `$deploy/sas-bases/examples/security/README.md` (for Markdown format) or at `$deploy/sas-bases/docs/configure_network_security_and_encryption_using_sas_security_certificate_framework.htm` (for HTML format).

Waiting for End of Discovery Error Message

Error

The following error message is displayed:

```
waiting for end of discovery after bootstrap
```

Explanation

During a failover event, a pod that was originally classified as the primary pod for the Crunchy Data operator can be demoted to a replica. If that occurs, when the Crunchy Data operator is later updated, the database tries to restart as a replica. The result is the error message.

Resolution

1 Find out the PostgreSQL pod name.

```
kubectl -n name-of-namespace get pods | grep postgres
```

2 Change the replica back to the primary pod.

Note: The command should be entered as a single line.

```
kubectl -n name-of-namespace exec -it postgresSQL-pod-name -c database
-- pg_ctl promote -D /pgdata/sas-crunchy-data-postgres
```

For *postgresSQL-pod-name*, use the pod name that was the output of step 1.

Certframe Container Fails at init Stage

Error

The certframe container on one or more pods fails with the following log entries:

```
[ERROR] - Certificate failed to show a status of 'Ready' with a new resource version
after retrying for 300 seconds
```

or

```
[ERROR] - Could not find a CertificateRequest in 'Ready=False' state associated with
Certificate
```

Explanation

A pod has not completed initialization, and the certframe init container is moving into a failed state.

Resolution

- 1 If a pod has not completed pod initialization due to the certframe container, search the logs of the certframe container for an error message such as the following:

```
[ERROR] - Could not find a CertificateRequest in 'Ready=False' state associated with
Certificate sas-risk-management-8d2d37cz42-p7rj9
```

- 2 If you find the error message, check the certificaterequests resource to determine whether there are multiple requests.

```
kubectl -n name-of-namespace --kubeconfig kubeconfig get certificaterequests
| grep search-term
```

For *search-term*, use part of the pod name. From the example error message, risk or management is a good search term. Any part of the pod name can be used, but the more uncommon the search term is, the fewer pod names there are to search through. Here is an example command that uses the sample error message in a namespace named test1:

```
kubectl -n test1 --kubeconfig kubeconfig get certificaterequests | grep risk
```

Here is sample output from the command:

sas-risk-management-app-8d2d37cz42-p7rj9-5xpr7	True	123m
sas-risk-management-app-8d2d37cz42-p7rj9-j6g5c	True	126m

- 3 If you find multiple certificaterequests, delete the oldest ones until only one is left.

```
kubectl -n name-of-namespace --kubeconfig kubeconfig delete
certificaterequest full-pod-name
```

Here is an example that uses output from the preceding steps.

```
kubectl -n test1 --kubeconfig kubeconfig delete certificaterequest sas-
risk-management-app-8d2d37cz42-p7rj9-j6g5c
```

- 4 Check the pod status.

Crunchy PostgreSQL Operator etcd Error Message

Note: If your deployment uses an external instance of PostgreSQL, skip this topic.

Error

When performing a SAS Viya deployment that is configured to use an internal instance of PostgreSQL, the PostgreSQL pods may not be created. If this scenario occurs, the following `kubectl get` command does not return any entries:

```
kubectl -n name-of-namespace get deployments -l crunchy-pgha-scope=sas-crunchy-data-postgres
```

Instead of entries, the following error message may be displayed in the `sas-crunchy-data-postgres-operator` pod:

```
etcdserver: leader changed
```

The error message may be embedded inside other error messages. The following is a sample error message containing the indicated error:

```
time="2022-02-04T15:45:40Z" level=error msg="error creating missing user secrets:
\"etcdserver: leader changed\""
func="internal/operator/cluster.AddClusterBase()" file="internal/operator/cluster/
cluster.go:137" version=4.7.3
```

Explanation

The etcd server of your Kubernetes cluster can restart or experience issues due to workloads on the cluster. If the etcd server is in an unstable state, the `sas-crunchy-data-postgres-operator` may fail to start up PostgreSQL.

Resolution

Recreate the PostgreSQL cluster so that the `sas-crunchy-data-postgres-operator` can try again.

- 1 Delete the existing PostgreSQL instance. Deletion will preserve the data, and the instance should be automatically recreated.

```
kubectl -n name-of-namespace delete pgcluster.crunchydata.com/sas-crunchy-data-postgres
```

- 2 Wait for PostgreSQL deployments to be created and then run the following command to confirm that the etcd server is running as expected.

```
kubectl -n name-of-namespace get deployments | grep sas-crunchy-data-postgres
```

Invalid Subject Name in Partner's Certificate Error Message

Note: This topic applies to deployments on Amazon Elastic Kubernetes Service only. If you are not deploying on Amazon Elastic Kubernetes Service, skip this topic.

Error

The following error message is displayed:

```
ERROR: A communication subsystem partner link setup request failure has occurred.
ERROR: Network request failed - SSL Error: Invalid subject name in partner's certificate.
Subject name must match machine name.
```

Explanation

Amazon Elastic Kubernetes Service uses a different format for host names than other supported providers. Currently, SAS Viya uses the format supported by the other providers. As a result, the server validation fails and the SAS/CONNECT client throws an error that prevents the deployment from working properly when you use a load balancer in the deployment.

Resolution

SAS is currently working on a fix for this issue. Until the fix is in place, use the following steps as a workaround.

- 1 Open the `$deploy/sas-bases/examples/security/customer-provided-merge-sas-certframe-configmap.yaml` file.
- 2 Find the `SAS_CERTIFICATE_ADDITIONAL_SAN_DNS=` variable in the literals block.
- 3 Add `*ec2.internal` to the value for this variable:

```
SAS_CERTIFICATE_ADDITIONAL_SAN_DNS=myhostalias.mydomain.com myhostalias myotheralias
*ec2.internal
```

Note: For more information about the `SAS_CERTIFICATE_ADDITIONAL_SAN_DNS=` variable, see the comments in the `$deploy/sas-bases/examples/security/customer-provided-merge-sas-certframe-configmap.yaml` file.

- 4 Re-deploy your SAS Viya software.

SAS Viya Deployment Operator Reconcile Failures

Error

After you apply the SASDeployment custom resource, checking the status results in a `FAILED` state.

Explanation

The state of the content modified for deployment, such as the `kustomization.yaml` file or any of the example YAML files used for configuration, is not what the SAS Viya Deployment Operator expects. The result is a conflict that causes the deployment to fail.

Resolution

See [“Initial Troubleshooting” on page 70](#) for information about determining the cause of failure. The `.status.messages` field contains the actual messages from Kustomize and may prove the most useful in diagnosing problems.

Performing Updates Produces Many Error Messages

Error

During the update process, you see many errors messages in the logs.

Explanation

A number of error messages come from timing issues that resolve themselves in the course of the update. Some might come from third-party software and are more like warnings or information messages. But since they originate with third-party software, SAS cannot change the content of the messages.

Resolution

The error messages should begin to diminish after 30 minutes. If more errors continue to appear in the logs, then you need to investigate.

Failure to Create Symbolic Link

SAS Configurator for Open Source requires a PVC for Python builds. If you want to use Microsoft Azure File Services (`azurefile`) as a storage class for the PVC, `msysmlinks` must be supplied as a mount option to the `azurefile` storage class in order to enable the update functionality. Here is an example:

```
kubectl describe storageclass azurefile -n production
Name: azurefile
IsDefaultClass: No
Annotations: none
Provisioner: kubernetes.io/azure-file
Parameters: skuName=Standard_LRS
AllowVolumeExpansion: True
MountOptions:
  mfsymlinks
actimeo=30 ReclaimPolicy: Delete
VolumeBindingMode: Immediate
Events: none
```

If the option is not supplied, a non-fatal message is generated in the SAS Configurator for Open Source pod log that reports a failure to create a symbolic link on the file system. The message provides the target of the link to be referenced. The target can be used to reference the Python install.

Long Delay for SAS/CONNECT Server Sign-on from SAS Studio

You might see a long (2–minute) delay during sign-on to a SAS/CONNECT server from an internal SAS Viya client, such as SAS Studio. You can work around this issue by setting `TCPNOIPADDR` to 1.

This problem has been observed on Microsoft Azure, AWS, and GCP. The delay is caused by an errant connection back to the SAS/CONNECT spawner. The spawner hangs until time-out, at which point a successful connection is made to the launched SAS/CONNECT server.

A fix is planned. In the meantime, run your programs with `TCPNOIPADDR=1` to work around the issue that causes the delay.

